BAB 4
HASIL PENELITIAN

4.1 RINGKASAN HASIL PENELITIAN

Penelitian ini berfokus pada perancangan dan implementasi sistem berbasis web untuk memfasilitasi *monitoring* tiket aduan gangguan layanan internet di PT. Lintas Data Prima. Sistem ini dikembangkan menggunakan *framework* Django sebagai kerangka kerja pengembangan aplikasi web dan MySQL sebagai sistem manajemen basis data relasional. Integrasi kedua teknologi ini memungkinkan pencatatan laporan aduan serta pemantauan progres penanganan tiket secara terstruktur dan *real-time*.

Fitur-fitur utama yang diimplementasikan dalam sistem ini mencakup tampilan *dashboard* yang menyajikan ringkasan data operasional, fungsionalitas pencarian dan pengelolaan kontak pelanggan, mekanisme penugasan tiket kepada teknisi, serta kapabilitas pengelolaan laporan aduan. Pemilihan MySQL didasarkan pada kemampuannya untuk mendukung pengelolaan data dalam skala besar dengan stabilitas tinggi. Berdasarkan serangkaian pengujian fungsionalitas menggunakan metode *black-box testing* dari total 45 kasus uji, 35 kasus uji (77,78%) berhasil lulus dan 10 kasus uji (22,22%) tidak lulus. Hasil ini mengindikasikan bahwa sebagian besar fungsionalitas utama sistem beroperasi sesuai dengan tujuan yang ditetapkan, yaitu mempercepat proses penanganan aduan gangguan internet dan meningkatkan efisiensi operasional PT. Lintas Data Prima.

4.2 IMPLEMENTASI DESAIN ANTARMUKA

Implementasi desain antarmuka sistem *monitoring* tiket pelayanan aduan gangguan internet dilakukan dengan menggunakan Django sebagai kerangka kerja utama. Django dipilih karena mendukung pengembangan aplikasi web yang cepat, aman, dan terstruktur. Sistem ini dirancang dengan menerapkan arsitektur *Model-Template-View (MTV)*, yang merupakan adaptasi dari pola *Model-View-Controller (MVC)*. Dalam penerapannya, logika tampilan ditulis menggunakan pendekatan *Class-Based View (CBV)* untuk meningkatkan keteraturan, keterbacaan, dan

efisiensi kode. Berikut desain antarmuka pada sistem *monitoring* pelayanan aduan gangguan internet di PT. Lintas Data Prima berbasis Aplikasi Web TicketApp.

4.2.1 Halaman *Dashboard* Admin

Tampilan *Dashboard* Admin pada gambar menyajikan ringkasan data sistem secara keseluruhan, berbeda dengan *dashboard* pelanggan yang hanya menampilkan tiket milik pengguna tersebut. Admin dapat melihat total jumlah tiket yang masuk, termasuk jumlah tiket yang masih terbuka, sudah ditutup, maupun yang belum diproses. Selain itu, *dashboard* juga menampilkan total pengguna berdasarkan peran seperti Admin, Teknisi, dan Pelanggan. Di bagian bawah, terdapat daftar tiket terbaru dari seluruh pengguna. Antarmuka ini dirancang untuk membantu admin memantau aktivitas sistem secara menyeluruh dan mempermudah pengelolaan tiket dan pengguna.

Gambar 4.1 Halaman *Dashboard* Admin

Berikut potongan *code* untuk menampilkan halaman *dashboard* admin.

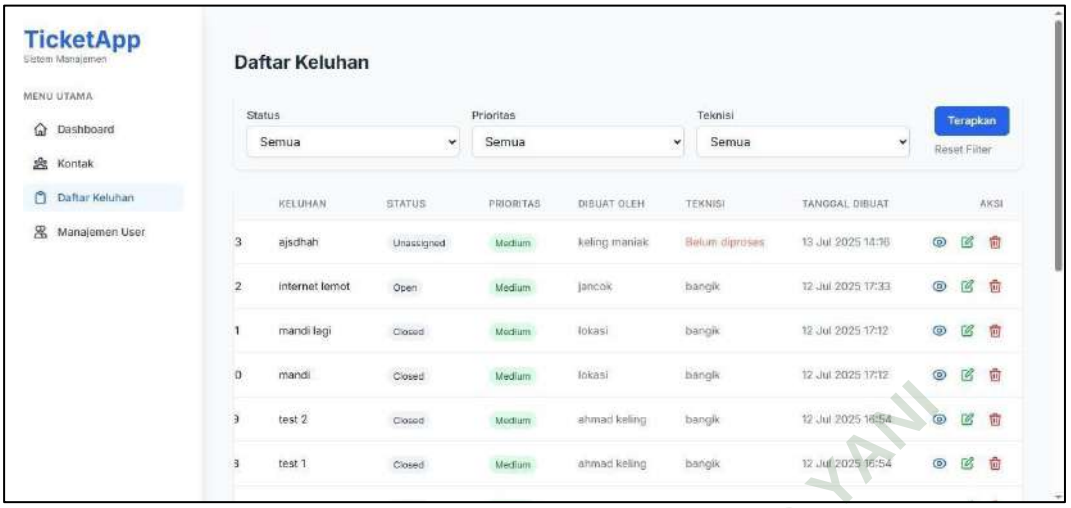
```
class AdminDashboardView(
    LoginRequiredMixin, AdminOnlyMixin, TemplateView):

    template_name = 'admin/dashboard.html'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['total_users'] = User.objects.count()
        context['admins'] = User.objects.filter(
            role='Admin').count()
        context['teknisi'] = User.objects.filter(
```

```
            role='Teknisi').count()
        context['pelanggan'] = User.objects.filter(
            role='Pelanggan').count()
        context['perusahaan'] = User.objects.filter(
            role='Perusahaan').count()
        context['total_tickets'] = Ticket.objects.count()
        context['tickets_open'] = Ticket.objects.filter(
            status='Open').count()
        context['tickets_closed'] = Ticket.objects.filter(
            status='Closed').count()
        context['tickets_unassigned'] = Ticket.objects.filter(
            status='Unassigned').count()
        context['tickets_high'] = Ticket.objects.filter(
            priority='High').count()
        context['tickets_medium'] = Ticket.objects.filter(
            priority='Medium').count()
        context['tickets_low'] = Ticket.objects.filter(
            priority='Low').count()
        context['recent_tickets'] = Ticket.objects.order_by(
            '-timestamp')[:5]
        context['recent_contacts'] = Contact.objects.order_by(
            '-timestamp')[:5]
        return context
```

Kelas Admin*DashboardView* pada Django digunakan untuk menampilkan *dashboard* administrator dengan membatasi akses hanya bagi pengguna yang telah *login* dan memiliki peran admin. Melalui metode *get_context_data*(), sistem mengumpulkan data berupa jumlah total dan klasifikasi pengguna, status serta prioritas tiket layanan, serta lima entri terbaru dari tiket dan kontak. Data tersebut dikirim ke templat untuk ditampilkan secara real-time sebagai dukungan pemantauan sistem oleh administrator.

4.2.2 Tampilan Daftar Keluhan Admin

Tampilan daftar keluhan admin pada sistem ticketapp menyajikan fitur penyaringan berdasarkan status, prioritas, dan teknisi, serta tabel yang memuat data keluhan secara rinci, termasuk kolom keluhan, status, prioritas, pembuat, teknisi, tanggal pembuatan, dan aksi. Seluruh tiket memiliki prioritas *medium*, dengan mayoritas telah ditangani teknisi, dan satu keluhan masih berstatus *unassigned*. Aksi seperti melihat, mengedit, dan menghapus dapat dilakukan langsung melalui ikon pada tabel. Antarmuka ini dirancang untuk mendukung efisiensi pengelolaan tiket oleh admin.



Gambar 4.2 Halaman Daftar Keluhan

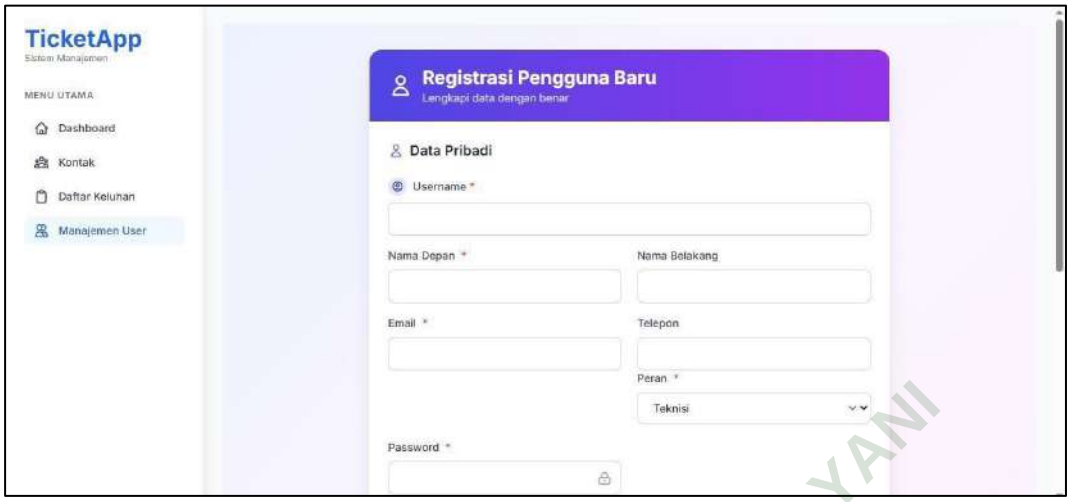
Berikut potongan *code* untuk menampilkan halaman Daftar Keluhan.

```
class TicketListView(LoginRequiredMixin, ListView):
    model = Ticket
    template_name = 'ticket/ticket_list.html'
    context_object_name = 'tickets'
    ordering = ['-timestamp']
```

TicketListView adalah class-based *view* Django yang menampilkan daftar tiket dari model *Ticket* untuk pengguna yang sudah *login*, menggunakan template *ticket_list*.html, dengan konteks bernama *ticket*s, dan mengurutkan tiket berdasarkan timestamp terbaru.

4.2.3 Halaman Membuat *User*

Tampilan Registrasi Pengguna Baru pada sistem TicketApp menyajikan formulir *input* data pribadi. Seluruh kolom bertanda bintang wajib diisi, dengan pilihan peran tersedia dalam bentuk *dropdown*. Antarmuka ini dirancang sederhana dan responsif untuk memudahkan proses penambahan akun pengguna ke dalam sistem.



Gambar 4.3 Halaman Membuat *User*

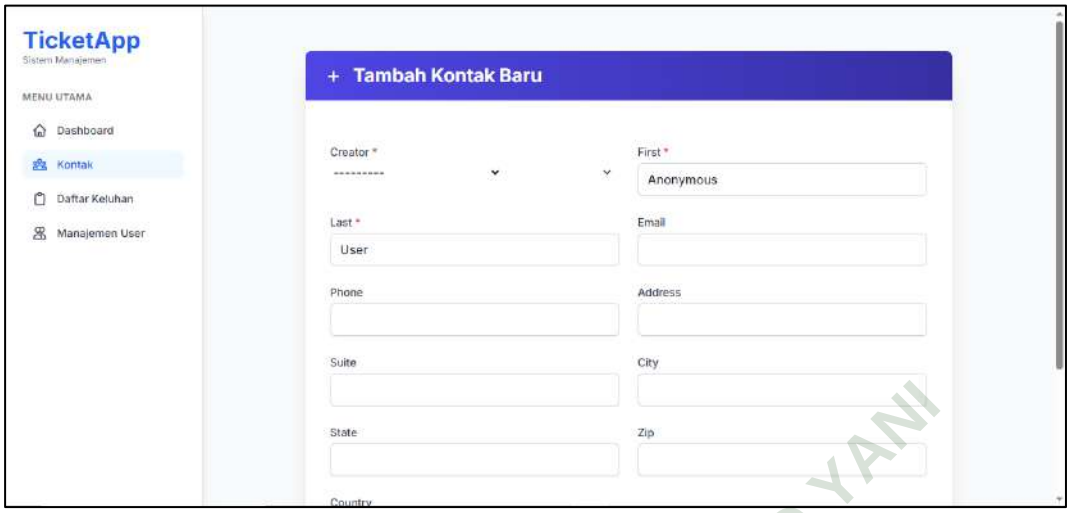
Berikut potongan *code* untuk menampilkan halaman membuat *user.*

```
class AdminUserCreateView(
    LoginRequiredMixin,
    AdminOnlyMixin,
    CreateView
):
    model = User
    form_class = AdminUserCreationForm
    template_name = (
        'admin/admin_user_form.html'
    )
    success_url = reverse_lazy(
        'admin-user-list'
    )
```

Admin*UserCreateView* adalah *view* dari admin untuk membuat *user* baru menggunakan *form* Admin*UserCreationForm*, hanya bisa diakses oleh admin yang *login*, dan setelah berhasil akan diarahkan ke halaman daftar *user*.

4.2.4 Halaman Membuat Kontak

Halaman Tambah Kontak Baru digunakan oleh admin untuk mencatat data kontak pengguna ke dalam sistem. Formulir ini berisi isian seperti pembuat, nama depan, nama belakang, surel, nomor telepon, alamat, kota, provinsi, kode pos, dan negara. Beberapa kolom wajib diisi ditandai dengan tanda bintang merah (*). Dengan adanya halaman ini, admin dapat menyimpan informasi kontak secara lengkap untuk keperluan tindak lanjut layanan atau komunikasi dengan pengguna.



Gambar 4.4 Halaman Membuat Kontak

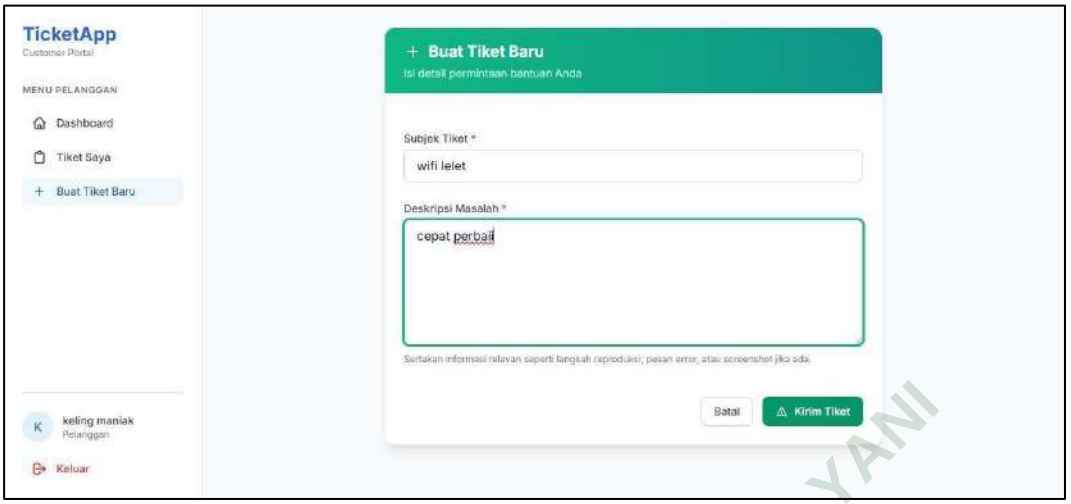
Berikut potongan *code* untuk menampilkan halaman membuat kontak

```
class ContactCreateView(LoginRequiredMixin, CreateView):
    model = Contact
    form_class = ContactForm
    template_name = 'contact/contact_form.html'
    success_url = reverse_lazy('contact-list')

    def form_valid(self, form):
        form.instance.creator = self.request.user
        return super().form_valid(form)
```

ContactCreateView adalah *view* Django untuk menambah data kontak, hanya bisa diakses oleh admin. Data yang dikirim lewat *ContactForm* akan disimpan dengan otomatis mencatat siapa pengguna yang membuatnya, lalu diarahkan ke halaman daftar kontak.

4.2.5 Halaman Membuat Tiket Pelanggan

Halaman "Buat Tiket Baru" pada aplikasi TicketApp memungkinkan pelanggan untuk mengirimkan laporan atau permintaan bantuan kepada tim dukungan. Pengguna diminta mengisi subjek tiket, seperti "wifi lelet", dan deskripsi masalah secara jelas, misalnya "cepat perbaiki". Setelah itu, mereka dapat mengirimkan tiket dengan menekan tombol "Kirim Tiket". Halaman ini juga dilengkapi menu navigasi di sisi kiri untuk mengakses *dashboard*, daftar tiket, dan tombol keluar akun.



Gambar 4.5 Halaman Membuat Tiket

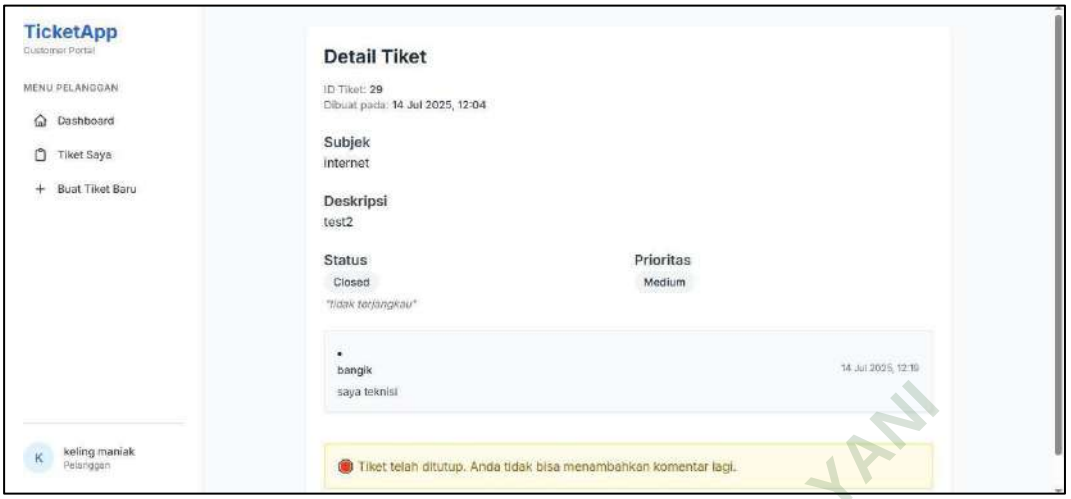
Berikut potongan kode untuk menampilkan pembuatan kontak baru:

```
class PelangganTicketCreateView(PelangganOnlyMixin, CreateView):
    model = Ticket
    fields = ['subject', 'body']
    template_name = 'pelanggan/ticket_form.html'
    success_url = reverse_lazy('pelanggan_ticket_list')

    def form_valid(self, form):
        form.instance.creator = self.request.user
        form.instance.owner = None
        return super().form_valid(form)
```

Pelanggan*TicketCreateView* adalah *view* Django untuk pelanggan membuat tiket baru. Menggunakan *CreateView*, *form* menampilkan *field* subject dan *body*, lalu menyimpan tiket dengan *creator sebag*ai *user* yang *login* dan *owner* diset ke *None*, sebelum mengalihkan ke daftar tiket.

4.2.6 Halaman Detail Tiket Pelanggan

Halaman Detail Tiket di TicketApp menampilkan informasi tiket seperti subjek, deskripsi, status, dan prioritas. Jika statusnya "*Closed*", akan muncul keterangan dari teknisi di bawah ikon status, seperti "tidak terjangkau", serta notifikasi bahwa tiket telah ditutup dan tidak bisa dikomentari lagi.



Gambar 4.6 Halaman Detail Tiket Pelanggan

Berikut potongan kode untuk menampilkan detail tiket

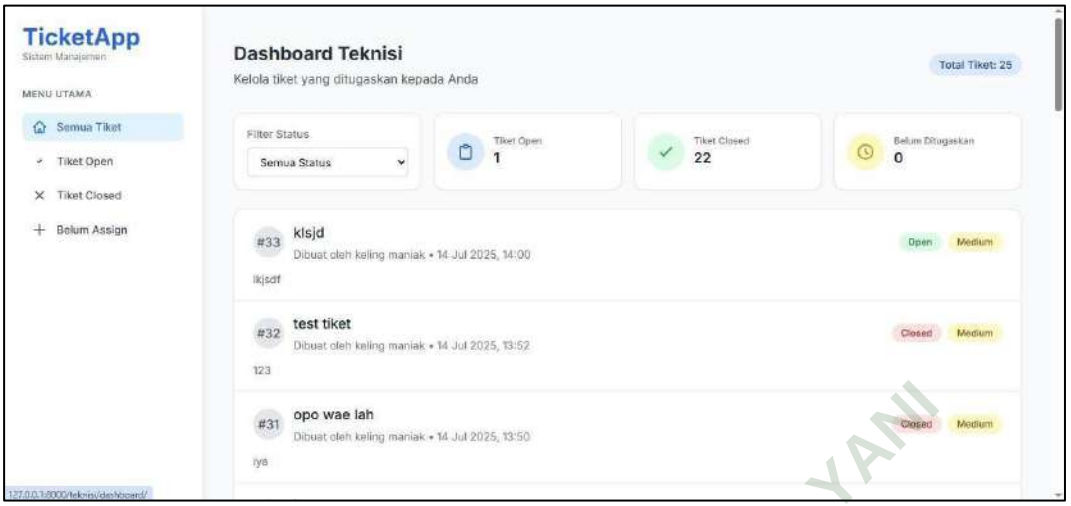
```
class PelangganTicketDetailView(DetailView):
    model = Ticket
    template_name = 'pelanggan/ticket_detail.html'

    def get_queryset(self):
        # Agar hanya bisa akses tiket miliknya sendiri
        return Ticket.objects.filter(
            creator=self.request.user
        )

    def get_context_data(self, **kwargs):
        context = super().get_context_data(
            **kwargs
        )
        ticket = self.object

        closing_comment = None
        if ticket.status == 'Closed':
            comments = ticket.comments.filter(
                creator_role_in=[
                    'Admin', 'Teknisi'
                ],
                is_closing=True
            ).order_by('-timestamp')

            if comments.exists():
                closing_comment = comments.first()

        context['form'] = CommentForm()
        context['closing_comment'] = (
            closing_comment
        )
        return context
```

```
    def post(self, request, *args, **kwargs):
        self.object = self.get_object()
        ticket = self.object
        form = CommentForm(request.POST)

        if form.is_valid():
            comment = form.save(
                commit=False
            )
            comment.creator = request.user
            comment.ticket = ticket
            comment.save()
            messages.success(
                request,
                "Komentar berhasil ditambahkan."
             )
        else:
            messages.error(
                request,
                "Terjadi kesalahan saat menambahkan "
                "komentar."
            )

        return redirect(
            'pelanggan_ticket_detail',
            pk=ticket.pk
        )
```

Pelanggan*Ticket*Detail*View* adalah *view* Django yang menampilkan detail tiket milik pelanggan, termasuk *form* komentar. *View* ini membatasi akses hanya ke tiket yang dibuat oleh pengguna *login*. Jika tiket sudah ditutup, komentar penutup dari admin/teknisi juga ditampilkan. Saat *form* komentar dikirim (*POST*), komentar baru akan disimpan dan ditautkan ke tiket tersebut.

4.2.7 Halaman *Dashboard* Teknisi

Halaman *Dashboard* Teknisi di TicketApp menampilkan ringkasan tiket yang ditugaskan kepada teknisi, termasuk jumlah tiket yang *Open*, *Closed*, dan Belum Ditugaskan. Jika terdapat tiket berstatus *Open*, berarti tiket tersebut telah ditugaskan oleh admin kepada teknisi dan siap untuk ditangani. Teknisi dapat melihat detail tiket seperti nomor, subjek, pengirim, tanggal, serta prioritasnya langsung dari daftar ini.



Gambar 4.7 Halaman *Dashboard* Teknisi

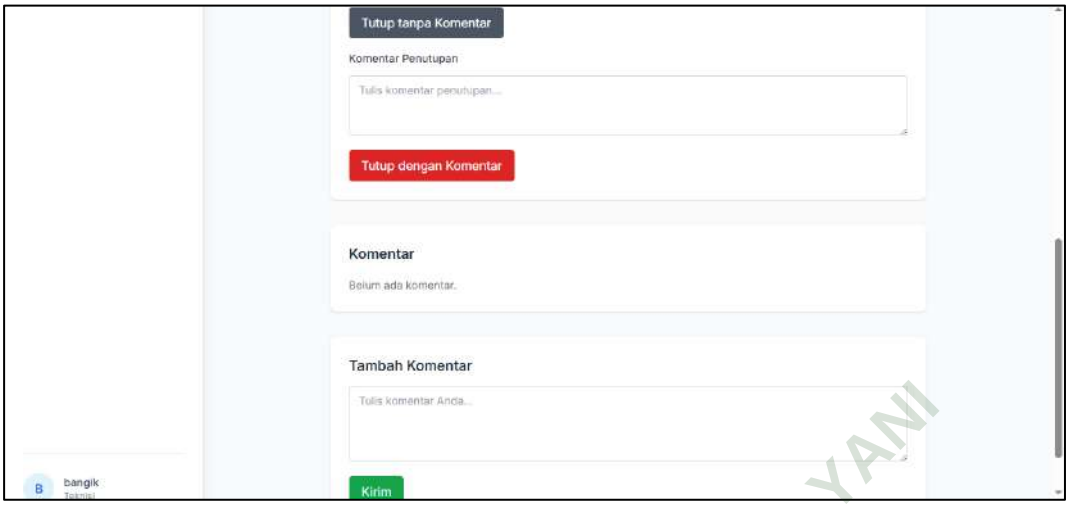
```
class TeknisiDashboardView(
    LoginRequiredMixin,
    TeknisiRequiredMixin,
    ListView
):
    model = Ticket
    template_name = 'teknisi/dashboard.html'
    context_object_name = 'tickets'
    login_url = reverse_lazy('login')

    def get_queryset(self):
        queryset = Ticket.objects.filter(
            owner=self.request.user
        ).order_by('-timestamp')

        status = self.request.GET.get('status')

        if status in dict(
            Ticket.STATUS_CHOICES
        ).keys():
            queryset = queryset.filter(
                status=status
            )

        return queryset

    def get_context_data(self, **kwargs):
        context = super().get_context_data(
            **kwargs
        )

        teknisi = self.request.user

        all_tickets = Ticket.objects.filter(
            owner=teknisi
        ).order_by('-timestamp')
```

```
        context['current_status'] = self.request.GET.get(
            'status', ''
        )

        context['open_count'] = all_tickets.filter(
            status='Open'
        ).count()

        context['closed_count'] = all_tickets.filter(
            status='Closed'
        ).count()

        context['unassigned_count'] = Ticket.objects.filter(
            owner_isnull=True
        ).count()

        return context
```

Teknisi*DashboardView* adalah *view* Django yang menampilkan daftar tiket milik teknisi yang sedang *login*. *View* ini memungkinkan penyaringan tiket berdasarkan status melalui *query* parameter, dan menampilkan data statistik seperti jumlah tiket berstatus *Open*, *Closed*, serta tiket yang belum ditugaskan (*unassigned*). Data tersebut ditampilkan dalam template teknisi/*dashboard*.html, dengan konteks tambahan seperti status yang sedang dipilih dan jumlah masing-masing kategori tiket.

4.2.8 Halaman *Update* Status Teknisi

Halaman penugasan teknisi di TicketApp menampilkan dua opsi penutupan tiket: “Tutup tanpa Komentar” dan “Tutup dengan Komentar”. Jika teknisi memilih tutup tanpa komentar, maka di halaman pelanggan tiket hanya ditandai sebagai *Closed* tanpa keterangan tambahan. Sebaliknya, jika teknisi menggunakan tutup dengan komentar dan mengisi kolom komentar penutupan, maka di halaman pelanggan akan muncul keterangan alasan penutupan di bawah status tiket.



Gambar 4.8 Halaman *Update* Status Teknisi

```
class UpdateTicketStatusView(
    LoginRequiredMixin,
    TeknisiRequiredMixin,
    View
):
    login_url = reverse_lazy('login')

    def post(self, request, pk):
        ticket = get_object_or_404(
            Ticket,
            pk=pk,
            owner=request.user
        )
        new_status = request.POST.get('status')
        comment_message = request.POST.get(
            'comment', ''
        ).strip()

        if new_status in dict(
            Ticket.STATUS_CHOICES
        ).keys():
            ticket.status = new_status
            ticket.save()
            messages.success(
                request,
                f"Status tiket diubah menjadi "
                f"{new_status}."
            )

            if comment_message:
                Comment.objects.create(
                    ticket=ticket,
                    creator=request.user,
                    message=comment_message,
                    is_closing=(new_status == 'Closed')
                )
```

```
        else:
            messages.error(
                request,
                "Status tidak valid."
            )

        return redirect(
            'ticket_detail',
            pk=pk
        )
```

UpdateTicketStatusView merupakan *view* yang dirancang khusus bagi pengguna dengan peran teknisi untuk memperbarui status tiket pelanggan melalui metode *POST*. *View* ini dilengkapi dengan validasi yang memastikan hanya teknisi yang telah melakukan *login* dan merupakan pemilik tiket (*owner = request.user*) yang memiliki hak untuk melakukan perubahan status. Jika status yang dikirim valid, maka status tiket akan diperbarui dan disimpan dalam basis data. Apabila teknisi menyertakan komentar, sistem akan menambahkan komentar tersebut ke tiket yang bersangkutan dan menandainya sebagai komentar penutup apabila status tiket berubah menjadi *"Closed"*. Setelah proses selesai, pengguna akan diarahkan kembali ke halaman detail tiket terkait.

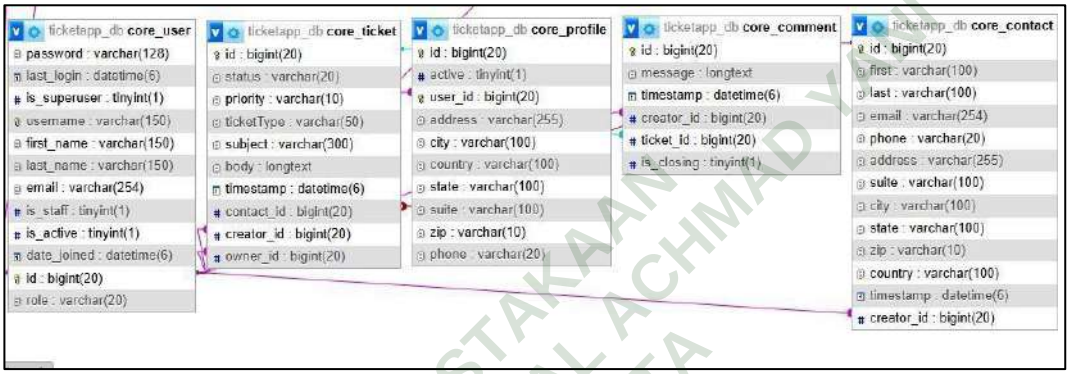
4.3 BASIS DATA

Basis data yang digunakan dalam sistem TicketApp adalah MySQL dan diintegrasikan dengan *framework* Django. Basis data ini berfungsi untuk menyimpan dan mengelola data pengguna, tiket aduan, serta aktivitas yang berkaitan dengan proses layanan. Sistem ini mendukung alur pelaporan, penugasan, dan penyelesaian gangguan secara efisien dan terstruktur, sehingga mempermudah pihak administrator, teknisi, maupun pelanggan dalam menjalankan peran dan tanggung jawabnya secara optimal.

4.3.1 Relasi Tabel

Basis data TicketApp terdiri atas beberapa tabel yang saling terhubung untuk mendukung sistem pengelolaan tiket layanan. Tabel *core_user* menyimpan data pengguna internal dan memiliki relasi satu-ke-banyak dengan tabel *core_ticket* melalui atribut *owner_id* dan *creator_id,* yang menunjukkan pengguna yang membuat dan memiliki tiket. Tabel *core_profile* berelasi satu-ke-satu dengan

core_user melalui *user_id*, menyimpan informasi tambahan seperti alamat dan nomor telepon. Tabel *core_ticket* juga berelasi dengan *core_contact* melalui *contact_id*, yang menyimpan data pelanggan eksternal terkait. Selain itu, tabel *core_comment* menghubungkan *core_user* dan *core_ticket* melalui *creator_id* dan *ticket_id* untuk mencatat komentar pengguna terhadap tiket. Relasi ini membentuk sistem yang terstruktur untuk mengelola tiket, identitas pengguna, informasi kontak, serta komunikasi yang terjadi selama proses penanganan tiket.



Gambar 4.9 Relasi Tabel

4.3.2 Struktur Tabel

Dari hasil analisis *database*, terdapat beberapa tabel utama dalam sistem. Salah satunya adalah tabel *ticket_user* yang menyimpan data pengguna. Tabel ini memuat informasi *login*, identitas pengguna, serta status dan hak aksesnya.

1. Tabel *User*

   Nama Tabel: *coret_user*

   Jumlah Kolom: 12

   Primary Key: id

Tabel 4.1 Tabel *User*

Field	***Type***
id	bigint(20)
password	varchar(128)
last_login	datetime(6)
is_superuser	tinyint(1)
username	varchar(150)

first_name	varchar(150)
last_name	varchar(150)
email	varchar(254)
is_staff	tinyint(1)
is_active	tinyint(1)
date_joined	datetime(6)
role	varcharr(20)

2. Tabel *Ticket*

   Nama Tabel: *core_ticket*

   Jumlah Kolom: 10

   Primary Key: id

Tabel 4.2 Tabel *Ticket*

Field	***Type***
id	bigint(20)
status	varchar(20)
priority	varchar(10)
ticketType	varchar(50)
subject	varchar(300)
body	longtext
timestamp	datetime(6)
contact_id	bigint(20)
creator_id	bigint(20)
owner_id	bigint(20)

3. Tabel *Comment*

   Nama Tabel: *core_Comment*

   Jumlah Kolom: 5

   Primary Key: id

Tabel 4.3 Tabel *Comment*

Field	***Type***

id	bigint(20)
message	longtext
timestamp	datetime(6)
creator_id	bigint(20)
ticket_id	bigint(20)

4. Tabel *Profile*

   Nama Tabel: *core_profile*

   Jumlah Kolom: 10

   Primary Key: id

Tabel 4.4 Tabel *Profile*

Field	***Type***
id	bigint(20)
active	tinyint(1)
user_id	bigint(20)
address	varchar(255)
city	varchar(100)
country	varchar(100)
state	varchar(100)
suite	varchar(100)
zip	varchar(10)
phone	varchar(20)

5. Tabel *Contact*

   Nama Tabel: *core_contact*

   Jumlah Kolom: 13

   Primary Key: id

Tabel 4.5 Tabel *Contact*

Field	***Type***
id	bigint(20)
first	varchar(100)

last	varchar(100)
email	varchar(254)
phone	varchar(20)
address	varchar(255)
suite	varchar(100)
city	varchar(100)
state	varchar(100)
zip	varchar(10)
country	varchar(100)
timestamp	datetime(6)
creator_id	bigint(20)
id	bigint(20)

4.4 FITUR-FITUR SISTEM

Sistem informasi pelayanan aduan gangguan internet yang dikembangkan dengan nama TicketApp menyediakan sejumlah fitur utama yang dibedakan berdasarkan peran pengguna, yaitu admin, teknisi, dan pelanggan. Setiap peran memiliki hak akses terhadap fitur tertentu yang telah disesuaikan dengan fungsi dan tanggung jawabnya dalam alur kerja sistem. Fitur-fitur tersebut dijabarkan sebagai berikut:

4.4.1 Fitur Untuk Admin

Admin memiliki peran pengelola sistem dan kontrol penuh terhadap data pengguna serta aduan yang masuk. Fitur-fitur yang dapat diakses oleh admin antara lain:

1. Tampilan *Dashboard* Admin

   Admin memiliki akses ke halaman *dashboard* yang menyajikan ringkasan data seperti jumlah tiket masuk, status tiket (terbuka, diproses, ditutup), jumlah teknisi aktif, serta grafik distribusi aduan berdasarkan kategori. Informasi ini ditampilkan secara visual sebagai pendukung proses pengambilan keputusan.
2. Membuat dan Mengelola Kontak

Admin dapat menambahkan informasi kontak yang bersifat administratif atau operasional dan digunakan dalam proses pelaporan dan penanganan aduan.

3. Membuat dan Mengelola *User*

   Sistem memungkinkan admin untuk membuat akun pengguna baru melalui formulir *input*, termasuk menetapkan peran sebagai pelanggan atau teknisi.

4. Mengelola Tiket

   Admin memiliki akses penuh untuk melihat, mengedit, menugaskan, atau menghapus tiket dari seluruh pengguna tanpa batasan.

5. Menugaskan Teknisi

   Admin dapat menetapkan teknisi tertentu untuk menangani aduan berdasarkan jenis masalah, lokasi, atau beban kerja teknisi.

6. Menambahkan Komentar Tiket

   Admin dapat memberikan komentar langsung pada tiket sebagai arahan, klarifikasi, atau pencatatan progres aduan.

4.4.2 Fitur Untuk Teknisi

Teknisi memiliki peran dalam menangani aduan pelanggan yang telah ditugaskan oleh admin. Sistem menyediakan fitur-fitur berikut:

1. *Dashboard* Teknisi

   Teknisi dapat melihat ringkasan tiket yang ditugaskan, status penyelesaian, dan notifikasi terkait tiket baru atau komentar. Tampilan ini membantu teknisi memahami beban kerja dan progres penyelesaian aduan.

2. Mengoperasikan Tiket yang Ditugaskan

   Teknisi dapat membuka tiket yang ditugaskan untuk membaca detail laporan, mengakses riwayat komentar, dan memperbarui status penanganan.

3. Menutup Tiket

   Teknisi memiliki opsi untuk menutup tiket setelah proses penanganan selesai. Terdapat dua kondisi penutupan tiket.

1. Menutup Tiket dengan Komentar

   Teknisi dapat menambahkan komentar saat menutup tiket. Komentar tersebut akan ditampilkan secara terbuka pada halaman detail tiket pelanggan sebagai dokumentasi teknis proses penanganan.

2. Menutup Tiket dengan Komentar

   Jika tiket ditutup tanpa menambahkan komentar, maka sistem hanya menampilkan ikon penutupan tiket tanpa informasi tambahan.

4.4.3 Fitur Untuk Pelanggan

Pelanggan adalah pengguna akhir yang melaporkan gangguan layanan internet. Fitur-fitur yang disediakan untuk pelanggan meliputi:

1. *Dashboard* Pelanggan

   Pelanggan memiliki akses ke halaman *dashboard* yang menampilkan ringkasan tiket yang telah dibuat, status tiket (belum ditangani, dalam proses, atau selesai), serta riwayat interaksi dengan teknisi. Halaman ini bertujuan memberikan transparansi dan kemudahan pemantauan aduan.

2. Membuat Tiket Baru

   Pelanggan dapat mengisi formulir pengaduan gangguan yang berisi jenis masalah, lokasi, dan deskripsi. Setelah dikirim, tiket akan tercatat dan menunggu proses verifikasi dari admin.

3. Memantau Tiket

   Pelanggan dapat melihat perkembangan status tiket, siapa teknisi yang menangani, dan waktu tanggapan terakhir.

4. Menambahkan Komentar

   Pelanggan dapat memberikan komentar tambahan jika diperlukan, baik untuk menjelaskan kembali masalah, menangg*api* teknisi, atau menyampaikan konfirmasi apabila masalah telah terselesaikan.

4.5 PENGUJIAN *BLACK BOX*

Black box testing adalah metode pengujian perangkat lunak yang menguji fungsi sistem dari sisi luar tanpa melihat struktur internal atau kode programnya. Pengujian ini dilakukan dengan memberikan *input* tertentu dan mengevaluasi

apakah *output* yang dihasilkan sesuai dengan spesifikasi sistem. Metode ini sangat efektif untuk menemukan kesalahan pada fungsi, antarmuka, dan validasi data, terutama pada aplikasi berbasis web (Wulandari et al., 2022) membuktikan efektivitas *black box testing* dengan teknik equivalence partitioning saat menguji sistem informasi akademik berbasis web, karena mampu mendeteksi kesalahan fungsional serta meningkatkan kualitas perangkat lunak yang dikembangkan.

4.5.1 Pengujian *Login*

Tabel 4.6 Pengujian *Black Box Login*

No	Fungsi Yang Diuji	Input	Langkah Pengujian	Ekspektasi Output	Hasil Uji
1	Login dengan Kredensial Valid	Nama Pengguna: admin Kata Sandi: admin123	Buka halaman login. Masukkan nama pengguna admin. Masukkan kata sandi admin123. Klik tombol Login.	Pengguna berhasil masuk ke sistem dan diarahkan ke halaman dasbor.	Berhasil
2	Login dengan Kredensial Tidak Valid	Nama Pengguna: user_unknown Kata Sandi: wrongpassword	Buka halaman login. Masukkan nama pengguna user_unknown. Masukkan kata sandi wrongpassword. Klik tombol Login.	Login gagal. Pengguna tetap di halaman login.	Berhasil
3	Login dengan Kata Sandi Tidak Valid	Nama Pengguna: user_valid Kata Sandi: wrongpassword	1. Buka halaman login.2. Masukkan nama pengguna user_valid. 3. Masukkan kata sandi wrongpassword. 4. Klik tombol Login.	Login gagal. Muncul pesan kesalahan "Nama pengguna atau kata sandi salah". Pengguna tetap di halaman login.	Gagal
4	Validasi Kolom Nama Pengguna Kosong	Nama Pengguna: (kosong)Kata Sandi: Password123	1. Buka halaman login.2. Biarkan kolom nama pengguna kosong.3. Masukkan kata sandi Password123. 4. Klik tombol Login.	Login gagal. Muncul pesan validasi "Nama pengguna tidak boleh kosong". Pengguna tetap di halaman login.	Gagal

5	Validasi Kolom Kata Sandi Kosong	Nama Pengguna: user_valid Kata Sandi: (kosong)	1. Buka halaman login.2. Masukkan nama pengguna user_valid. 3. Biarkan kolom kata sandi kosong. 4. Klik tombol Login.	Login gagal. Muncul pesan validasi "Kata sandi tidak boleh kosong". Pengguna tetap di halaman login.	Gagal

4.5.2 Pengujian Admin Membuat *User*

Tabel 4.7 Pengujian *black box* admin membuat *user*

No	Fungsi Yang Diuji	Input	Langkah Pengujian	Ekspektasi Output	Hasil Uji
1	Admin berhasil membuat user baru dengan data valid	Username: newuser1 Nama Depan: Budi Nama Belakang: Santoso Email: budi.santoso@example.com Telepon: 081234567890 Peran: Teknisi Password: Password123 Alamat Lengkap: Jl. Melati No. 10 Nomor Rumah: 10 Kota: Yogyakarta Provinsi: DI Yogyakarta Kode Pos: 55182 Negara: Indonesia	Admin masuk ke sistem. Navigasi ke halaman "Registrasi Pengguna Baru".Isi semua kolom dengan data valid seperti di input. Klik tombol "Simpan Data".	Pengguna baru berhasil terdaftar. Admin diarahkan ke daftar pengguna atau halaman sukses.	Berhasil
2	Validasi kolom wajib kosong (Pesan browser)	Username: (kosong) Nama Depan: (kosong) Email: (kosong) Peran: (kosong/default) Password: (kosong) Alamat Lengkap: (kosong) Kota: (kosong) Provinsi: (kosong) Kode Pos: (kosong) Negara: (kosong)	Admin masuk ke sistem. Navigasi ke halaman "Registrasi Pengguna Baru". Kosongkan semua kolom bertanda bintang (*). Klik tombol "Simpan Data".	Pembuatan user gagal. Pada kolom wajib pertama yang kosong, muncul tooltip atau pesan validasi bawaan peramban "Please fill out this field" (atau "Harap isi bidang ini"), dan pengiriman formulir dicegah.	Berhasil
3	Admin gagal membuat user (Email	Username: newuser2 Nama Depan: Citra Nama Belakang: Dewi Email: citra.dewi@invalid Telepon:	Admin masuk ke sistem. Navigasi ke	Pembuatan user gagal. Muncul keterangan validasi bahwa	Berhasil

	tidak valid format)	081234567891 Peran: Teknisi Password: Password123 Alamat Lengkap: Jl. Mawar No. 5 Nomor Rumah: 5 Kota: Surabaya Provinsi: Jawa Timur Kode Pos: 60111 Negara: Indonesia	halaman "Registrasi Pengguna Baru". Isi semua kolom dengan data valid, kecuali kolom Email dengan format tidak valid. Klik tombol "Simpan Data".	format Email tidak valid (misal: "Enter a valid email address").	
4	Admin gagal membuat user (Username sudah ada)	Username: newuser1 (sudah terdaftar di ADM-001) Nama Depan: Eka Nama Belakang: Putra Email: eka.putra@example.com Telepon: 081234567893 Peran: TeknisiPassword: Password456 Alamat Lengkap: Jl. Dahlia No. 1 Nomor Rumah: 1 Kota: Medan Provinsi: Sumatera Utara Kode Pos: 20111 Negara: Indonesia	Admin masuk ke sistem. Navigasi ke halaman "Registrasi Pengguna Baru". Isi semua kolom dengan data valid, kecuali Username yang sudah terdaftar. Klik tombol "Simpan Data".	Pembuatan user gagal. Muncul pesan kesalahan bahwa Username sudah digunakan (misal: "Username sudah ada, silakan gunakan username lain").	Gagal
5	Validasi semua kolom wajib kosong	Semua kolom bertanda bintang (*) kosong. Kolom opsional (Nama Belakang, Telepon, Nomor Rumah) juga kosong.	Admin masuk ke sistem. Navigasi ke halaman "Registrasi Pengguna Baru". Biarkan semua kolom bertanda bintang (*) kosong. Klik tombol "Simpan Data".	Pembuatan user gagal. Muncul pesan validasi untuk setiap kolom wajib yang kosong (misal: "Username tidak boleh kosong", "Nama Depan tidak boleh kosong", dll.).	Gagal

6	Admin berhasil membuat user dengan data minimal (hanya wajib)	Username: minuser Nama Depan: Fajar Email: fajar@example.com Peran: Teknisi Password: MinPassword1 Alamat Lengkap: Jln. Minimum Kota: Solo Provinsi: Jawa Tengah Kode Pos: 57111 Negara: Indonesia (Kolom lain dikosongkan/default)	Admin masuk ke sistem. Navigasi ke halaman "Registrasi Pengguna Baru". Isi hanya kolom yang bertanda bintang (*). Klik tombol "Simpan Data".	Pengguna baru berhasil terdaftar dengan data minimal. Muncul pesan keberhasilan.	Gagal
7	Admin gagal membuat user (Username sudah ada)	Username: newuser1 (sudah terdaftar di admin) Nama Depan: Eka Nama Belakang: Putra Email: eka.putra@example.com Telepon: 081234567893 Peran: TeknisiPassword: Password456 Alamat Lengkap: Jl. Dahlia No. 1 Nomor Rumah: 1 Kota: Medan Provinsi: Sumatera Utara Kode Pos: 20111 Negara: Indonesia	Admin masuk ke sistem. Navigasi ke halaman "Registrasi Pengguna Baru". Isi semua kolom dengan data valid, kecuali Username yang sudah terdaftar. Klik tombol "Simpan Data".	Pembuatan user gagal. Muncul keterangan kesalahan bahwa Username sudah digunakan (misal: " A user with that username already exists ").	Berhasil

4.5.3 Pengujian Admin Membuat Kontak

Tabel 4.8 Pengujian *black box* membuat kontak

No	Fungsi Yang Diuji	Input	Langkah Pengujian	Ekspektasi Output	Hasil Uji
1	Admin berhasil membuat kontak baru dengan data valid	Creator: Admin (pilih dari dropdown) First: John Last: Doe Email: john.doe@example.com Phone: 081234567890 Address: Jl. Contoh No. 1 Suite: A-1 City: Yogyakarta State: DI Yogyakarta Zip: 55123 Country: Indonesia	Admin masuk ke sistem. Navigasi ke halaman "Membuat Kontak". Isi semua kolom dengan data valid seperti di input. Klik	Kontak baru berhasil terdaftar. Admin diarahkan ke daftar kontak atau halaman sukses.	Berhasil

			tombol Simpan.		
2	Validasi kolom wajib Creator kosong (Pesan browser)	Creator: (kosong/tidak dipilih) First: Jane Last: Smith (Valid) Email, Phone, dll.: (Valid atau kosong)	Admin masuk ke sistem. Navigasi ke halaman "Membuat Kontak". Biarkan kolom Creator kosong/tidak dipilih. Isi kolom First dan Last dengan data valid. Klik tombol Simpan.	Pembuatan kontak gagal. Pada kolom Creator, muncul tooltip atau pesan validasi bawaan peramban "Please fill out this field" (atau "Harap isi bidang ini").	Berhasil
3	Validasi kolom wajib First kosong (Pesan browser)	Creator: Admin (Valid) First: (kosong) Last: Smith (Valid) Email, Phone, dll.: (Valid atau kosong)	Admin masuk ke sistem. Navigasi ke halaman "Membuat Kontak". Isi kolom Creator dan Last dengan data valid. Biarkan kolom First kosong. Klik tombol Simpan.	Pembuatan kontak gagal. Pada kolom First, muncul tooltip atau pesan validasi bawaan peramban "Please fill out this field" (atau "Harap isi bidang ini").	Berhasil
4	Validasi kolom wajib Last kosong (Pesan browser)	Creator: Admin (Valid) First: Jane (Valid) Last: (kosong) Email, Phone, dll.: (Valid atau kosong)	Admin masuk ke sistem. Navigasi ke halaman "Membuat Kontak". Isi kolom Creator dan First dengan data valid. Biarkan kolom Last kosong. Klik tombol Simpan.	Pembuatan kontak gagal. Pada kolom Last, muncul tooltip atau pesan validasi bawaan peramban "Please fill out this field" (atau "Harap isi bidang ini").	Berhasil
5	Validasi semua kolom wajib kosong (Pesan browser)	Creator: (kosong/tidak dipilih) First: (kosong) Last: (kosong) Email, Phone, dll.: (Valid atau kosong)	Admin masuk ke sistem. Navigasi ke halaman "Membuat Kontak". Kosongkan	Pembuatan kontak gagal. Pada kolom wajib pertama yang kosong (Creator), muncul tooltip	Berhasil

			semua kolom wajib (Creator, First, Last). Klik tombol Simpan.	atau pesan validasi bawaan peramban "Please fill out this field" (atau "Harap isi bidang ini"). Pengiriman formulir dicegah hingga diisi.	

4.5.4 Pengujian Admin Mengubah Status

Tabel 4.9 Pengujian *black box* admin mengubah status

No	Fungsi Yang Diuji	Input	Langkah Pengujian	Ekspektasi Output	Hasil Uji
1	Admin berhasil menugaskan teknisi pada tiket	Status: Open (tetap/valid) Priority: Medium (tetap/valid) Owner: bangik (Teknisi) (pilih teknisi spesifik)	Admin masuk ke sistem. Navigasi ke halaman "Update Tiket #33". Pastikan Status dan Priority terisi valid. Pilih teknisi bangik (Teknisi) dari dropdown Owner. Klik tombol Simpan Perubahan. Kembali ke halaman daftar tiket atau buka kembali tiket #33 untuk verifikasi.	Perubahan berhasil disimpan. Tiket #33 sekarang menampilkan bangik (Teknisi) sebagai Owner. Tidak ada pesan di halaman update tiket (sesuai spesifikasi).	Berhasil
2	Admin berhasil mengubah penugasan teknisi pada tiket	Status: Open (tetap/valid) Priority: Medium (tetap/valid) Owner: Teknisi B (pilih teknisi lain yang valid, misalkan "Teknisi A" ke "Teknisi B")	Admin masuk ke sistem. Navigasi ke halaman "Update Tiket #33". Pastikan Status dan Priority terisi valid. Ubah pilihan teknisi di dropdown Owner dari teknisi sebelumnya ke Teknisi B. Klik tombol Simpan Perubahan. Kembali ke halaman daftar tiket atau buka kembali tiket #33 untuk verifikasi.	Perubahan berhasil disimpan. Tiket #33 sekarang menampilkan Teknisi B sebagai Owner. Tidak ada pesan di halaman update tiket.	Berhasil

3	Admin berhasil menghapus penugasan teknisi (jika dropdown mengizinkan pilihan kosong/null)	Status: Open (tetap/valid) Priority: Medium (tetap/valid) Owner: (opsi "Tidak Ditugaskan" / kosong / null)	Admin masuk ke sistem. Navigasi ke halaman "Update Tiket #33". Pastikan Status dan Priority terisi valid. Pilih opsi yang mengindikasikan tanpa penugasan (jika ada, seperti "Tidak Ditugaskan" atau biarkan kosong jika memungkinkan) di dropdown Owner. Klik tombol Simpan Perubahan. Kembali ke halaman daftar tiket atau buka kembali tiket #33 untuk verifikasi.	Perubahan berhasil disimpan. Tiket #33 sekarang tidak menampilkan teknisi yang ditugaskan (atau status unassigned). Tidak ada pesan di halaman update tiket.	Gagal
4	Membatalkan perubahan penugasan teknisi	Status: Open (tetap/valid) Priority: Medium (tetap/valid) Owner: bangik (Teknisi) (pilih teknisi spesifik)	Admin masuk ke sistem. Navigasi ke halaman "Update Tiket #33". Pastikan Status dan Priority terisi valid. Pilih teknisi bangik (Teknisi) dari dropdown Owner. Klik tombol Kembali ke Daftar (atau tombol batal lainnya). Kembali ke halaman daftar tiket atau buka kembali tiket #33 untuk verifikasi.	Perubahan tidak disimpan. Tiket #33 tetap dengan Owner sebelumnya. Admin diarahkan kembali ke daftar tiket.	Gagal

4.5.5 Pengujian Admin Daftar Keluhan

Tabel 4.10 Pengujian *Black Box* Admin Daftar Keluhan

No	Fungsi Yang Diuji	*Input*	Langkah Pengujian	Ekspektasi *Output*	Hasil Uji
1	Admin berhasil melihat daftar keluhan	N/A	Admin masuk ke sistem. Navigasi ke halaman "Daftar Keluhan" melalui menu *sidebar*.	Halaman daftar keluhan berhasil dimuat. Tabel menampilkan semua kolom: ID, Keluhan, Status, Prioritas,	Berhasil

				Dibuat Oleh, Teknisi, Tanggal Dibuat, Aksi. Data keluhan yang ada di sistem ditampilkan dengan benar.	
2	Verifikasi tampilan data keluhan yang ada	N/A	Admin masuk ke sistem. Navigasi ke halaman "Daftar Keluhan". Periksa setidaknya 3 baris data keluhan yang berbeda (ID, Keluhan, Status, Prioritas, Dibuat Oleh, Teknisi, Tanggal Dibuat) apakah sesuai dengan data yang diharapkan ada di *database*.	Setiap baris data keluhan dalam tabel menampilkan informasi yang akurat dan konsisten dengan data yang tersimpan.	Berhasil
3	Verifikasi fungsionalitas tombol "Lihat Detail" (ikon mata)	N/A	Admin masuk ke sistem. Navigasi ke halaman "Daftar Keluhan". Klik ikon "mata" pada salah satu baris keluhan (misal: ID #33).	Admin diarahkan ke halaman detail keluhan untuk tiket yang dipilih (ID #33). Informasi detail keluhan ditampilkan dengan benar pada halaman tersebut.	Berhasil
4	Verifikasi fungsionalitas tombol "Edit" (ikon pensil)	N/A	Admin masuk ke sistem. Navigasi ke halaman "Daftar Keluhan". Klik ikon "pensil" pada salah satu baris keluhan	Admin diarahkan ke halaman *form* edit keluhan untuk tiket yang dipilih (ID #33), dengan semua kolom terisi	Berhasil

			(misal: ID #33).	data tiket yang akan diedit.	
5	Verifikasi tampilan keluhan dengan status berbeda	N/A	Admin masuk ke sistem. Navigasi ke halaman "Daftar Keluhan". Periksa tampilan keluhan dengan status "*Open*", "*Closed*", dan "*Unassigned*" (jika ada).	Keluhan dengan berbagai status (*Open*, *Closed*, *Unassigned*) ditampilkan dengan label atau indikator visual yang sesuai dan benar.	Berhasil

4.5.6 Pengujian Penugasan Teknisi

Tabel 4.11 Pengujian *black box* penugasan teknisi

No	Fungsi Yang Diuji	Input	Langkah Pengujian	Ekspektasi Output	Hasil Uji
1	Teknisi berhasil melihat tiket yang ditugaskan (Tiket Open)	N/A	Teknisi masuk ke sistem menggunakan kredensialnya (bangik). Navigasi ke dashboard teknisi atau klik menu "Tiket Open" di sidebar.	Halaman dashboard teknisi berhasil dimuat. Daftar tiket menampilkan tiket-tiket dengan status "Open" yang ditugaskan kepada teknisi tersebut (misal: #33, #32). Ringkasan "Tiket Open" (2) dan "Total Tiket" (2) menampilkan jumlah yang sesuai.	Berhasil
2	Verifikasi detail tiket terbuka di dashboard	N/A	Teknisi masuk ke sistem. Navigasi ke dashboard teknisi. Periksa informasi yang ditampilkan untuk tiket #33 dan #32 (ID,	Informasi setiap tiket yang ditampilkan di daftar (ID, Keluhan, Dibuat Oleh, Tanggal Dibuat, Status, Prioritas) sesuai	Berhasil

			Keluhan, Dibuat Oleh, Tanggal Dibuat, Status, Prioritas).	dengan data yang diharapkan.	
3	Teknisi berhasil membuka detail tiket terbuka	N/A	Teknisi masuk ke sistem. Navigasi ke dashboard teknisi. Klik pada salah satu tiket terbuka di daftar (misal: tiket #33 "klsjd").	Teknisi berhasil diarahkan ke halaman detail tiket #33. Semua informasi rinci tentang tiket tersebut (termasuk deskripsi lengkap, riwayat, dll., jika ada) ditampilkan dengan benar.	Berhasil
4	Filter status "Open" di dashboard	N/A	Teknisi masuk ke sistem. Navigasi ke dashboard teknisi. Pastikan dropdown "Filter Status" menunjukkan "Open" sebagai default atau pilih "Open".	Daftar tiket hanya menampilkan tiket-tiket dengan status "Open" yang ditugaskan kepada teknisi tersebut.	Berhasil
5	Verifikasi jumlah "Tiket Closed" dan "Belum Ditugaskan"	N/A	Teknisi masuk ke sistem. Navigasi ke dashboard teknisi. Periksa nilai pada card "Tiket Closed" dan "Belum Ditugaskan".	Nilai pada card "Tiket Closed" (20) dan "Belum Ditugaskan" (0) harus akurat sesuai dengan jumlah tiket yang relevan di sistem untuk teknisi tersebut.	Gagal

4.5.7 Pengujian Detail Tiket Teknisi

Tabel 4.12 Pengujian *black box* detail tiket teknisi

No	Fungsi Yang Diuji	Input	Langkah Pengujian	Ekspektasi Output	Hasil Uji
1	Teknisi berhasil menutup tiket	N/A	Teknisi masuk ke sistem. Navigasi ke halaman detail	Status tiket #33 berubah menjadi "Closed". Tidak	Berhasil

	tanpa komentar		tiket yang open (misal: ID #33). Klik tombol "Tutup tanpa Komentar". Kembali ke daftar tiket atau buka kembali tiket #33 untuk verifikasi.	ada komentar penutupan yang ditambahkan. Tidak ada notifikasi langsung di halaman detail tiket.	
2	Teknisi berhasil menutup tiket dengan komentar	Komentar Penutupan: Masalah sudah diselesaikan.	Teknisi masuk ke sistem. Navigasi ke halaman detail tiket yang open (misal: ID #32). Ketikkan komentar "Masalah sudah diselesaikan." di kolom "Tulis komentar penutupan...". Klik tombol "Tutup dengan Komentar". Kembali ke daftar tiket atau buka kembali tiket #32 untuk verifikasi.	Status tiket #32 berubah menjadi "Closed". Komentar "Masalah sudah diselesaikan." berhasil ditambahkan ke detail tiket pelanggan Tidak ada notifikasi langsung di halaman detail tiket.	Berhasil
3	Teknisi berhasil menambahkan komentar baru ke tiket	Tulis Komentar Anda: Perlu pengecekan lebih lanjut.	Teknisi masuk ke sistem. Navigasi ke halaman detail tiket (misal: ID #33, baik open maupun closed). Ketikkan komentar "Perlu pengecekan lebih lanjut." di kolom "Tulis komentar Anda...". Klik tombol Kirim. Refresh halaman atau buka kembali tiket #33 untuk verifikasi.	Komentar "Perlu pengecekan lebih lanjut." berhasil ditambahkan ke bagian "Komentar". Tidak ada notifikasi langsung di halaman detail tiket.	Berhasil

4	Mengirim komentar kosong	Tulis Komentar Anda: (kosong)	Teknisi masuk ke sistem. Navigasi ke halaman detail tiket. Biarkan kolom "Tulis komentar Anda..." kosong. Klik tombol Kirim.	Komentar tidak ditambahkan. Tidak ada notifikasi langsung di halaman detail tiket, tetapi server tidak menyimpan komentar.	Berhasil
5	Verifikasi perubahan status tiket setelah ditutup	N/A	Lakukan langkah ADM-UPD-001 atau ADM-UPD-002. Kembali ke Dashboard Teknisi atau Daftar Keluhan (menu Admin). Periksa status tiket yang baru saja ditutup.	Status tiket yang telah ditutup (misal: ID #33) di dashboard atau daftar keluhan berubah dari "Open" menjadi "Closed".	Gagal

4.5.8 Pengujian Buat Tiket Pelanggan

Tabel 4.13 Pengujian *black box* buat tiket pelanggan

No	Fungsi Yang Diuji	*Input*	Langkah Pengujian	Ekspektasi *Output*	Hasil Uji
1	Pelanggan berhasil membuat tiket baru dengan data valid	Subjek Tiket: Koneksi Internet Lemot Deskripsi Masalah: Internet di rumah saya sangat lambat sejak kemarin malam, tidak bisa membuka *website.*	*Pel*anggan masuk ke *Customer Port*al. Navigasi ke halaman "Buat Tiket Baru". Isi kolom Subjek Tiket dan Deskripsi Masalah dengan data valid. Klik tombol Kirim Tiket. Kembali ke halaman "Tiket Saya" atau *dashboard* untuk verifikasi.	Tiket baru berhasil dibuat dan tercatat di sistem. Tiket tersebut seharusnya muncul di daftar "Tiket Saya" pelanggan. Tidak ada pesan keberhasilan langsung di halaman "Buat Tiket Baru".	Berhasil
2	Pelanggan gagal membuat tiket	Subjek Tiket: (kosong) Deskripsi Masalah: Laptop saya tidak bisa menyala. (valid)	Pelanggan masuk ke *Customer Port*al. Navigasi	Pembuatan tiket gagal. Pada kolom Subjek Tiket, muncul	Berhasil

	(Subjek Tiket kosong)		ke halaman "Buat Tiket Baru". Biarkan kolom Subjek Tiket kosong. Isi kolom Deskripsi Masalah dengan data valid. Klik tombol Kirim Tiket.	*tooltip* atau pesan validasi bawaan peramban "*Please fill out this field*" (atau "Harap isi bidang ini").	
3	Pelanggan gagal membuat tiket (Deskripsi Masalah kosong)	Subjek Tiket: Printer Tidak Berfungsi (valid) Deskripsi Masalah: (kosong)	Pelanggan masuk ke *Customer Port*al. Navigasi ke halaman "Buat Tiket Baru". Isi kolom Subjek Tiket dengan data valid. Biarkan kolom Deskripsi Masalah kosong. Klik tombol Kirim Tiket.	Pembuatan tiket gagal. Pada kolom Deskripsi Masalah, muncul *tooltip* atau pesan validasi bawaan peramban "*Please fill out this field*" (atau "Harap isi bidang ini").	Berhasil
4	Pelanggan gagal membuat tiket (Kedua kolom wajib kosong)	Subjek Tiket: (kosong) Deskripsi Masalah: (kosong)	Pelanggan masuk ke *Customer Port*al. Navigasi ke halaman "Buat Tiket Baru". Biarkan kedua kolom wajib (Subjek Tiket dan Deskripsi Masalah) kosong. Klik tombol Kirim Tiket.	Pembuatan tiket gagal. Pada kolom Subjek Tiket, muncul *tooltip* atau pesan validasi bawaan peramban "*Please fill out this field*" (atau "Harap isi bidang ini"). Pengiriman formulir dicegah hingga kolom tersebut diisi.	Berhasil

4.5.9 Pengujian *Dasboard* Pelanggan

Tabel 4.14 Pengujian *black box dasboard* pelanggan

No	Fungsi Yang Diuji	*Input*	Langkah Pengujian	Ekspektasi *Output*	Hasil Uji
1	Pelanggan berhasil	N/A	Pelanggan masuk ke aplikasi dengan	Halaman *Dashboard*	Berhasil

	mengakses *Dashboard* Pelanggan		kredensial valid (keling maniak). Verifikasi halaman yang dimuat setelah *login*.	Pelanggan berhasil dimuat. Ucapan "Selamat Datang, keling maniak" dan informasi ringkasan tiket ditampilkan dengan benar.	
2	Verifikasi ringkasan jumlah tiket	N/A	Pelanggan berada di *Dashboard* Pelanggan. Periksa nilai pada *card* "Total Tiket", "Dalam Proses", dan "Selesai".	Nilai pada *card* ringkasan tiket harus akurat dan sesuai dengan jumlah tiket yang dimiliki pelanggan.	Berhasil
3	Verifikasi informasi profil pelanggan	N/A	Pelanggan berada di *Dashboard* Pelanggan. Periksa informasi profil yang ditampilkan (nama, *email*, dan tanggal bergabung).	Informasi profil pelanggan (nama, *email*, dan tanggal bergabung) ditampilkan dengan benar dan akurat.	Berhasil
4	Navigasi dari tombol "Buat Tiket Baru" (*Dashboard*)	N/A	Pelanggan berada di *Dashboard* Pelanggan. Klik tombol + Buat Tiket Baru di kanan atas halaman.	Pelanggan berhasil diarahkan ke halaman *form* "Buat Tiket Baru".	Berhasil
5	Navigasi dari menu *sidebar* "Tiket Saya"	N/A	Pelanggan berada di *Dashboard* Pelanggan. Klik menu "Tiket Saya" di *sidebar*.	Pelanggan berhasil diarahkan ke halaman yang menampilkan daftar semua tiket yang dimilikinya.	Berhasil

Hasil pengujian black-box yang telah dilaksanakan menunjukkan bahwa dari total 45 kasus uji yang dieksekusi, 35 kasus uji (77,78%) berhasil lulus dan 10 kasus uji (22,22%) tidak lulus. Data ini mengindikasikan bahwa sebagian besar fungsionalitas utama sistem beroperasi sesuai dengan spesifikasi yang diharapkan. Namun, ditemukannya 10 kasus uji yang gagal mengindikasikan adanya defek pada beberapa fitur esensial, yang memerlukan identifikasi lebih lanjut serta perbaikan untuk menjamin stabilitas dan keandalan sistem secara keseluruhan.

4.6 PEMBAHASAN

Bagian ini menguraikan analisis terhadap implementasi sistem *monitoring* tiket pelayanan aduan gangguan internet yang telah dikembangkan, serta mengevaluasi kontribusinya dalam mengatasi permasalahan yang teridentifikasi di PT. Lintas Data Prima. Fokus pembahasan meliputi peningkatan efisiensi proses pelaporan, percepatan respons penanganan aduan, dan peningkatan transparansi dalam alur kerja internal.

Sistem ini, yang diimplementasikan menggunakan framework Django dan basis data MySQL, telah melalui proses pengujian fungsionalitas dengan metode black-box testing. Hasil pengujian menunjukkan bahwa dari total 45 kasus uji yang dieksekusi, 35 kasus uji (77,78%) berhasil lulus dan 10 kasus uji (22,22%) tidak lulus. Data ini mengindikasikan bahwa sebagian besar fungsionalitas utama sistem beroperasi sesuai dengan spesifikasi yang diharapkan. Namun, ditemukannya 10 kasus uji yang gagal mengindikasikan adanya defek pada beberapa fitur esensial, yang memerlukan identifikasi lebih lanjut serta perbaikan untuk menjamin stabilitas dan keandalan sistem secara keseluruhan. Transformasi dari proses manual menjadi digital melalui sistem ini diharapkan dapat memberikan dampak signifikan terhadap peningkatan efektivitas operasional perusahaan.

Berikut ini adalah pembahasan lebih lanjut yang dikelompokkan menjadi dua bagian utama.

4.6.1 Peningkatan Efisiensi dan Akuntabilitas Penanganan Aduan

Sebelum implementasi sistem ini, proses pelaporan gangguan di PT. Lintas Data Prima dilakukan secara konvensional melalui saluran komunikasi verbal (telepon) atau aplikasi perpesanan. Metode ini memiliki potensi risiko tinggi terhadap kehilangan informasi serta *delay* dalam proses penanganan. Dengan diimplementasikannya sistem *monitoring* tiket berbasis digital, proses pelaporan menjadi lebih terstruktur dan efisien, di mana setiap aduan pelanggan langsung tercatat sebagai tiket dalam sistem.

Fitur penugasan teknisi secara langsung oleh administrator terbukti mempercepat alur distribusi pekerjaan dan memberikan kejelasan informasi

mengenai penanggung jawab tiket. Selain itu, seluruh aktivitas terkait tiket, mulai dari inisiasi hingga penyelesaian, secara otomatis terdokumentasi dalam sistem. Hal ini menciptakan rekam jejak yang komprehensif, yang dapat dimanfaatkan untuk keperluan audit, evaluasi kinerja, dan peningkatan layanan berkelanjutan. Oleh karena itu, sistem ini berkontribusi secara signifikan dalam mempercepat penanganan keluhan pelanggan, meningkatkan akuntabilitas, dan pada akhirnya, berdampak positif terhadap kepuasan pelanggan terhadap layanan yang diberikan.

4.6.2 Optimalisasi Pengalaman Pengguna dan Kemudahan Operasional

Desain antarmuka sistem yang dikembangkan dengan pendekatan sederhana dan intuitif memberikan dampak substansial terhadap *usability*. Pemanfaatan HTML dan CSS *Tailwind* untuk desain *responsive* memastikan kemudahan navigasi dan aksesibilitas sistem pada berbagai perangkat, baik bagi administrator, teknisi, maupun pelanggan. Setiap peran pengguna diberikan akses terhadap menu dan fungsionalitas yang relevan dengan tanggung jawab masing- masing, meminimalkan kompleksitas dan potensi kesalahan operasional.

Halaman *dashboard* menyediakan agregasi informasi penting, seperti jumlah tiket aktif, prioritas, dan status tiket, yang memfasilitasi pengambilan keputusan cepat oleh administrator. Teknisi dapat secara langsung melihat tiket yang ditugaskan kepada mereka dan memperbarui status penanganan tanpa mengalami kesulitan yang berarti. Desain yang berorientasi pada pengguna ini tidak hanya mempercepat proses adaptasi pengguna terhadap sistem, tetapi juga secara efektif mengurangi tingkat kesalahan operasional. Dengan demikian, sistem ini dapat dioperasikan secara optimal oleh berbagai pihak tanpa memerlukan pelatihan yang kompleks, mendukung kelancaran operasional harian PT. Lintas Data Prima.