BAB III
PELAKSANAAN PROYEK KEMANUSIAAN

3.1 Tugas dan Uraian Kerja

Sebagai seorang volunteer dalam proyek pengembangan Chatbot Edukatif untuk Mitigasi Bencana di Lebak Selatan, saya berperan aktif dalam merancang dan mengembangkan chatbot yang bertujuan untuk memberikan informasi kebencanaan yang mudah diakses oleh masyarakat. Proyek ini merupakan bagian dari inisiatif Gugus Mitigasi Lebak Selatan (GMLS) untuk meningkatkan literasi kebencanaan di wilayah pesisir Kabupaten Lebak.

Selama berperan sebagai volunteer, saya terlibat langsung dalam tahap perencanaan dan pengembangan chatbot, dimulai dengan identifikasi kebutuhan informasi yang relevan mengenai mitigasi bencana, jalur evakuasi, serta peringatan dini. Kegiatan ini mencakup penyusunan alur percakapan chatbot yang sederhana

dan mudah dipahami oleh masyarakat, serta pengumpulan dan pengintegrasian konten edukatif mengenai berbagai jenis bencana alam seperti tsunami dan gempa bumi.

Selain itu, saya juga terlibat dalam tahap pengujian prototipe chatbot, untuk memastikan chatbot berfungsi dengan baik dan memberikan informasi yang akurat kepada masyarakat. Uji coba ini mencakup analisis terhadap umpan balik pengguna dan penyempurnaan alur percakapan agar lebih interaktif dan responsif terhadap pertanyaan pengguna.

Setelah chatbot selesai, saya turut berkontribusi dalam penyuluhan kepada masyarakat mengenai cara menggunakan chatbot untuk mendapatkan informasi kebencanaan dengan mudah. Selain itu, saya bekerja sama dengan tim GMLS untuk memastikan bahwa chatbot tetap diperbarui dengan informasi terbaru dan relevan seiring dengan perubahan kondisi bencana yang ada.

Melalui peran ini, saya dapat berkontribusi dalam meningkatkan ketangguhan masyarakat terhadap bencana, serta memastikan bahwa teknologi seperti chatbot dapat digunakan secara maksimal untuk memperkuat kesiapsiagaan bencana di wilayah rawan bencana seperti Lebak Selatan.

Berikut adalah uraian kerja selama saya melakukan humanity project

No	Tahapa Kegiatan	Uraian Pekerjaan	Waktu Pelaksanaan	Lama Pengerjaan
1	Kunjungan Awal, Pengamatan, dan Menentukan Ide Proyek	- Kunjungan pertama ke lokasi - Mengenal tim GMLS dan kegiatan mereka - Mengamati kondisi dan menyepakati projek	17 - 27 Februari	11 Hari
2	Mengumpulkan Materi Edukasi Bencana dan Membuat Prototype Awal	- Mengambil beberapa informasi mengenai bencana dari situs GMLS dan beberapa situs lain - Menyusun materi yang mudah di pahami oleh masyarakat - Mengerjakan tampilan chatbot dengan HTML, CSS, JavaScript. - Menambahkan beberapa fitur	28 Februari - 13 April	46 Hari

3	Pertemuan Kedua dengan GMLS dan diskusi Flow Chatbot	- Kunjungan kedua ke lokasi. - Berusaha membahas alur percakapan chatbot dengan pihak GMLS. - Tetapi belum mencapai final karena komunikasi tidak lancar.	14 - 23 April	10 Hari
4	Menyempurnakan Alur Percakapan dan Mendesain	-Menyempurnakan isi dari percakapan chatbot	24 April - 18 Mei	25 Hari

	Ulang	- Menyesuaikan desain dengan website GMLS yang baru karena adanya pergantian desain website - Menambahkan beberapa fitur seperti deteksi lokasi dan juga chat langsung dengan pihak GMLS via Whatsapp		
5	Pertemuan Terakhir, Uji Coba, dan Evaluasi	-Menunjukkan chatbot kepada pihak GMLS - Meminta tanggapan dan evaluasi dari pihak GMLS	19 - 28 Mei	10 Hari
6	Perbaikan Akhir dan Implementasi ke Website GMLS	- Memperbaiki isi dan tampilan chatbot sesuai dengan hasil evaluasi dengan pihak gmls - Memasukkan chatbot ke dalam website GMLS	29 Mei - 12 Juni	15 Hari

Tabel 3.1 Tabel Uraian Kerja

3.1.1 Kunjungan Awal, Pengamatan, dan Menentukan Ide Proyek

Pada tahap awal, saya melakukan kunjungan langsung ke lokasi kegiatan di Kecamatan Panggarangan, Kabupaten Lebak. Kunjungan ini bertujuan untuk mengenal lebih dekat kondisi masyarakat dan struktur kerja GMLS sebagai mitra lokal.

Selama periode ini, saya mengikuti aktivitas komunitas, berdiskusi dengan anggota GMLS, dan mencatat berbagai tantangan yang mereka hadapi, khususnya dalam aspek edukasi kebencanaan. Melalui proses pengamatan ini, kami bersama-sama menyepakati bahwa kebutuhan utama masyarakat adalah akses informasi kebencanaan yang mudah dan cepat. Oleh karena itu, ide pengembangan chatbot edukatif dipilih sebagai bentuk kontribusi yang paling sesuai untuk proyek ini.

3.1.2 Mengumpulkan Materi Edukasi Bencana dan Membuat Prototype Awal

Setelah kesepakatan ide tercapai, tahap selanjutnya difokuskan pada pengumpulan materi kebencanaan yang akan digunakan dalam chatbot. Saya menelusuri berbagai sumber, seperti website resmi GMLS, dokumen BNPB, serta referensi ilmiah dari jurnal-jurnal mitigasi bencana.

Materi yang terkumpul kemudian disederhanakan agar bisa dipahami masyarakat umum. Di fase ini pula, saya mulai membangun prototipe awal chatbot menggunakan HTML, CSS, dan JavaScript. Struktur chatbot dirancang untuk menyampaikan informasi berbasis pilihan tombol (quick reply) dan dilengkapi dengan komponen interaktif dasar. Beberapa fitur awal seperti menu utama dan sub-topik mitigasi mulai diuji secara internal. Pertemuan Kedua dengan GMLS dan Diskusi Flow Chatbot

Pada kunjungan kedua ini, saya kembali ke lokasi untuk mempresentasikan kemajuan awal proyek dan mendiskusikan alur

percakapan chatbot dengan tim GMLS. Tujuannya adalah menyelaraskan konten chatbot dengan konteks lokal dan pengalaman lapangan GMLS.

Namun, pada tahap ini, komunikasi dengan pihak GMLS masih belum optimal karena kesibukan mereka dalam kegiatan lain. Hal ini menghambat proses validasi alur chatbot secara mendalam. Oleh karena itu, hasil pertemuan ini belum menghasilkan finalisasi percakapan, namun menjadi bahan refleksi untuk perbaikan selanjutnya.

3.1.3 Menyempurnakan Alur Percakapan dan Mendesain Ulang

Setelah Trip 2, saya mulai mengerjakan chatbot secara lebih aktif. Saya menyempurnakan struktur percakapan agar lebih sistematis dan responsif terhadap kebutuhan pengguna. Konten yang sebelumnya bersifat informatif ditingkatkan menjadi lebih interaktif, termasuk fitur seperti pemilihan jenis bencana, tips evakuasi, dan titik kumpul evakuasi.

Selain itu, saya menyesuaikan desain tampilan chatbot agar serasi dengan website baru GMLS yang sedang dalam proses redesain. Saya juga menambahkan fitur deteksi lokasi (geolokasi) dan integrasi tombol WhatsApp agar pengguna dapat menghubungi tim GMLS langsung dari chatbot.

3.1.4 Pertemuan Terakhir, Uji Coba, dan Evaluasi

Pada kunjungan terakhir, saya menyajikan versi hampir final dari chatbot kepada tim GMLS. Chatbot diuji secara langsung oleh perwakilan anggota GMLS. Hasil evaluasi menunjukkan bahwa chatbot telah mampu menyampaikan informasi dengan baik dan sangat membantu dalam memberikan pemahaman dasar tentang mitigasi bencana. Saran yang diberikan terutama berfokus pada penambahan sedikit fitur dan sedikit penyesuaian.

3.1.5 Perbaikan Akhir dan Implementasi ke Website GMLS

Berdasarkan hasil evaluasi pada Trip 3, saya melakukan perbaikan akhir pada isi dan tampilan chatbot. Saya merapikan struktur dialog, menyempurnakan respons terhadap pertanyaan pengguna, serta memastikan setiap fitur berfungsi dengan baik.

Pada tanggal 12 Juni 2025, chatbot secara resmi diintegrasikan ke dalam website GMLS. Setelah proses penanaman, saya melakukan pengujian akhir untuk memastikan integrasi berjalan lancar dan chatbot dapat diakses oleh publik tanpa hambatan.

3.2 Solusi dari permasalahan

Sebagai bagian dari upaya meningkatkan literasi kebencanaan di wilayah pesisir Kabupaten Lebak, chatbot edukatif berbasis web dikembangkan dengan tujuan untuk memberikan informasi mengenai mitigasi bencana secara cepat dan akurat. Dengan bantuan teknologi, chatbot ini memungkinkan masyarakat mengakses informasi terkait

bencana seperti tsunami, gempa bumi, jalur evakuasi, dan sistem peringatan dini dengan mudah dan praktis.

Alur Umum Chatbot:

1. Pengguna mengakses chatbot melalui ikon yang muncul di situs GMLS.

2. Chatbot akan meminta lokasi pengguna menggunakan geolokasi, yang memungkinkan chatbot memberikan informasi sesuai dengan lokasi pengguna.

3. Interaksi dimulai, dengan chatbot menawarkan pilihan informasi yang relevan mengenai bencana, jalur evakuasi, dan peringatan dini.

4. Chatbot memberikan informasi berbasis percakapan, memberikan jawaban dan opsi interaktif sesuai pertanyaan yang diajukan oleh pengguna.

5. Pengguna dapat mendapatkan informasi spesifik, seperti lokasi evakuasi terdekat, cara mitigasi bencana, dan lainnya.

Gambar 3.1 Alur/Flow Chatbot

Diagram alur chatbot Mitra GMLS menggambarkan bagaimana sistem memberikan informasi kebencanaan secara interaktif kepada pengguna melalui alur percakapan berbasis tombol. Percakapan dimulai ketika pengguna membuka chatbot dan menekan tombol “Mulai”. Setelah itu, sistem akan secara otomatis menentukan lokasi pengguna menggunakan fitur geolokasi berbasis browser. Informasi lokasi ini digunakan untuk menyesuaikan konten kebencanaan, seperti titik evakuasi yang relevan.

Setelah lokasi ditentukan, pengguna akan diperlihatkan pilihan jenis bencana yang ingin dipelajari, yaitu Gempa, Banjir, Tanah Longsor, atau Tsunami. Selain itu, tersedia pula menu khusus berjudul “Program GMLS yang Wajib Anda Ketahui” sebagai bagian dari kampanye edukasi komunitas. Jika pengguna memilih salah satu bencana, chatbot akan menampilkan submenu berisi beberapa jenis informasi yang dapat dipilih, yaitu: penjelasan tentang bencana tersebut (“Apa itu…”), cara mitigasi,

daftar kit darurat, informasi tempat evakuasi, akses ke chat langsung dengan tim GMLS melalui WhatsApp, opsi kembali ke menu bencana, atau mengakhiri percakapan.

Secara khusus, jika pengguna memilih jenis bencana Tsunami, maka akan muncul satu pilihan tambahan yaitu “Indikator Tsunami Ready”. Pilihan ini tidak tersedia untuk jenis bencana lain karena indikator tersebut memang dirancang khusus untuk wilayah pesisir dengan potensi tsunami tinggi. Semua informasi dalam chatbot disampaikan melalui balasan teks dari sistem, dan pengguna cukup memilih opsi yang ditampilkan untuk melanjutkan percakapan. Chatbot akan terus merespons hingga pengguna memilih untuk kembali ke menu awal atau mengakhiri chat.

Alur ini menunjukkan bahwa chatbot tidak hanya menyampaikan informasi satu arah, tetapi dirancang untuk mengarahkan pengguna memilih topik yang paling relevan bagi situasi mereka, sehingga memberikan pengalaman edukatif yang personal dan terfokus.

Berikut juga merupakan beberapa code yang saya gunakan dalam pengembangan chatbot dan hasilnya selama mengikuti kegiatan Humanity Project

1. Html

```
<div class="chat-popup" id="chatPopup" onclick="toggleChat()">
  <img src="https://gmls.org/wp-content/uploads/2025/06/Avatar_GMLS.png" alt="GMLS Avatar" style="width: 72px; height: 72px; border-radius: 50%;">
</div>

<div class="chat-container-popup" id="chatContainer" style="display: none;">
  <div class="chat-header">
    <img src="https://gmls.org/wp-content/uploads/2025/06/Logo-GMLS-scaled.png" alt="Logo GMLS kiri" class="logo logo-left" />
    <div class="header-text">Mitra GMLS</div>
    <img src="https://gmls.org/wp-content/uploads/2025/06/Logo-GMLS-scaled.png" alt="Logo GMLS kanan" class="logo logo-right" />
    <button class="close-btn" onclick="closeChat()">❌</button>
  </div>
  <div class="messages" id="chat"></div>
  <div class="quick-buttons" id="quickButtons"></div>
  <div class="input-area" id="inputArea"></div>
</div>
```

Gambar 3.2 Html

Struktur HTML chatbot Mitra GMLS dimulai pada baris 1 hingga 3, yang membentuk elemen pemicu utama untuk menampilkan chatbot. Pada baris ini terdapat sebuah dengan class="chat-popup" dan id="chatPopup" yang berfungsi sebagai ikon chatbot di pojok layar. Di dalamnya terdapat elemen <img> yang memuat gambar avatar berbentuk bulat dari GMLS (menggunakan border-radius: 50%) dengan ukuran 72px. Ketika pengguna mengklik gambar ini, fungsi toggleChat() akan dipanggil, yang secara dinamis akan menampilkan atau menyembunyikan jendela chat.

Selanjutnya, pada baris 5 hingga 15, terdapat elemen utama dengan class="chat-container-popup" dan id="chatContainer" yang menyimpan seluruh komponen tampilan percakapan chatbot. Awalnya, elemen ini disembunyikan (display: none) dan hanya ditampilkan ketika pengguna mengaktifkan chatbot melalui avatar di baris sebelumnya. Pada baris 6 hingga 11, terletak struktur header chatbot. Di baris 6, <div

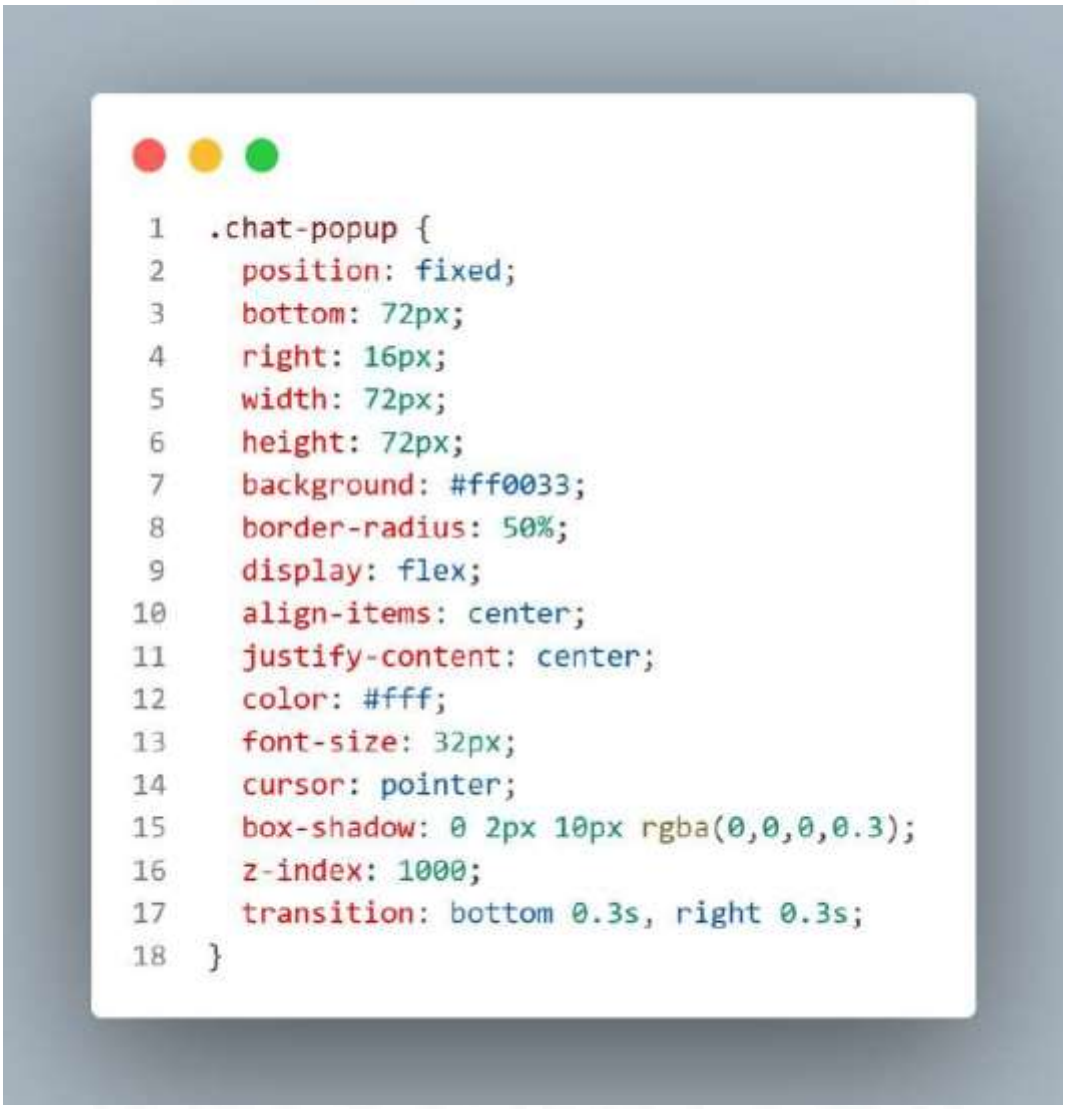
class="chat-header"> mengatur area kepala chatbot yang berisi tiga elemen utama: dua buah <img> yang menampilkan logo GMLS di sisi kiri (baris 7) dan kanan (baris 9), serta di baris 8 yang menampilkan teks “Mitra GMLS” sebagai identitas utama bot. Sementara itu, baris 10 menyematkan sebuah tombol bertanda □ yang akan menjalankan fungsi closeChat() saat diklik, yang akan menyembunyikan kembali chatbot.

Bagian isi percakapan dimulai pada baris 12, yaitu elemen dengan class="messages" dan id="chat". Inilah tempat seluruh riwayat percakapan antara pengguna dan chatbot ditampilkan. Pesan dari pengguna maupun dari chatbot akan dimasukkan ke dalam elemen ini menggunakan fungsi JavaScript seperti appendMessage().

Kemudian pada baris 13, terdapat dengan class="quick-buttons" dan id="quickButtons". Elemen ini digunakan untuk menampilkan tombol-tombol respons cepat atau pilihan menu yang akan muncul berdasarkan tahapan percakapan (seperti pilihan jenis bencana atau informasi mitigasi). Tombol-tombol ini diatur secara dinamis dengan JavaScript, dan menjadi metode utama interaksi pengguna dengan chatbot. Terakhir, pada baris 14, terdapat elemen dengan class="input-area" dan id="inputArea". Meskipun dalam versi saat ini area ini tidak digunakan untuk input teks langsung, namun ia disiapkan untuk fleksibilitas ke depan—misalnya untuk tombol “Mulai”, tampilan ulang chat, atau bahkan input bebas jika diimplementasikan.

Dengan struktur yang modular ini, setiap bagian dari chatbot dapat dikontrol dan diubah tampilannya secara terpisah melalui skrip JavaScript. Pendekatan ini menjadikan chatbot mudah untuk dikembangkan, dipelihara, dan diintegrasikan ke dalam halaman web komunitas seperti situs GMLS.

2. Tampilan Ikon Chatbot Awal



```css
.chat-popup {
  position: fixed;
  bottom: 72px;
  right: 16px;
  width: 72px;
  height: 72px;
  background: #ff0033;
  border-radius: 50%;
  display: flex;
  align-items: center;
  justify-content: center;
  color: #fff;
  font-size: 32px;
  cursor: pointer;
  box-shadow: 0 2px 10px rgba(0,0,0,0.3);
  z-index: 1000;
  transition: bottom 0.3s, right 0.3s;
}
```

Gambar 3.3 Tampilan Ikon Chatbot Awal

Penataan visual untuk ikon chatbot mengandalkan class.chat-popup. Properti position: fixed (baris 2) memastikan bahwa ikon akan selalu muncul menempel di layar bagian bawah kanan, walaupun pengguna menggulir halaman. Properti bottom: 72px dan right: 16px (baris 3–4) menentukan jarak dari tepi bawah dan kanan layar.

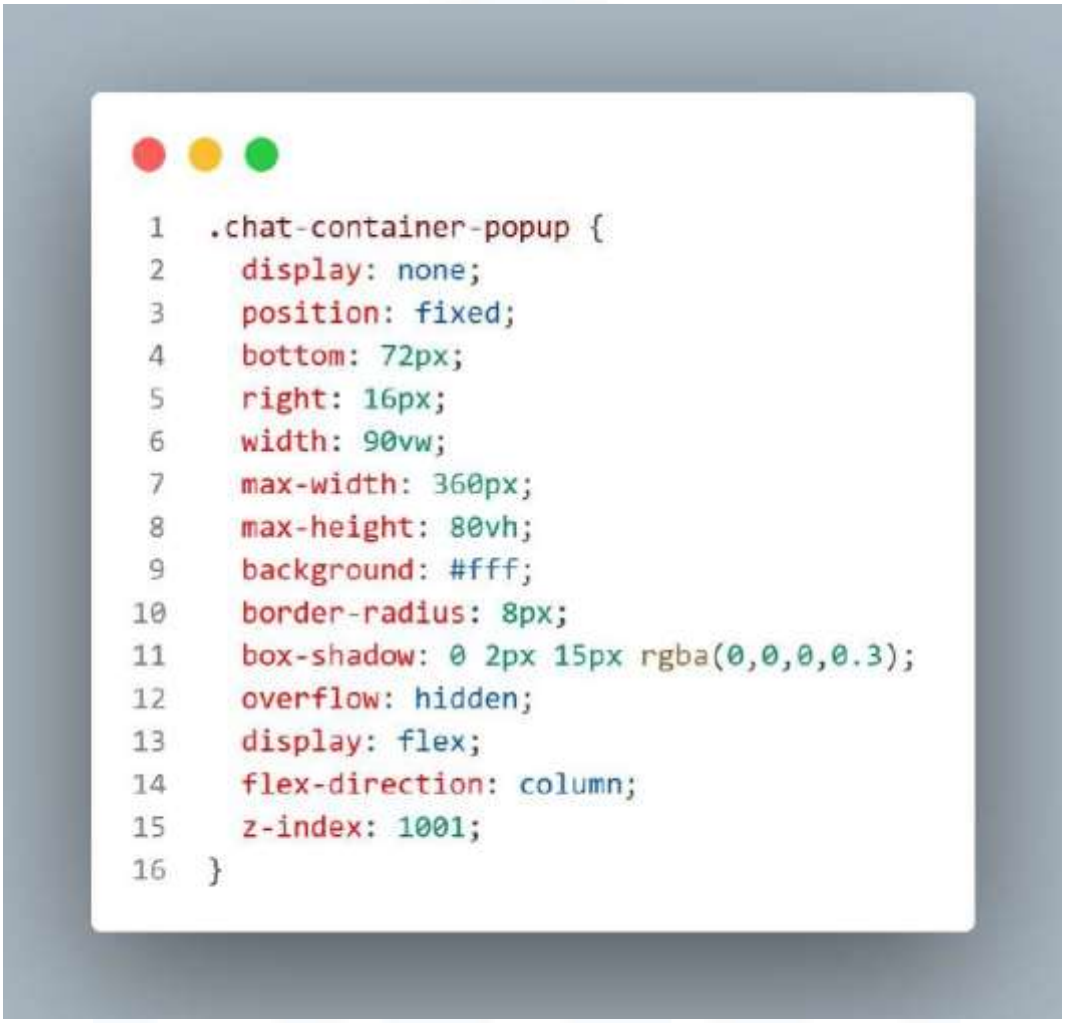
Selanjutnya, ukuran ikon diatur melalui width dan height masing-masing 72 piksel (baris 5–6). Warna latar belakang ditentukan melalui background: #ff0033 (baris 7), yang merupakan warna merah khas identitas visual GMLS. Bentuk ikon dibulatkan secara sempurna menggunakan border-radius: 50% (baris 8).

Dengan display: flex, seluruh isi div diatur agar berada di tengah menggunakan align-items: center dan justify-content: center (baris 9–11). Teks atau ikon di dalamnya akan muncul di posisi sentral. Warna teks (jika ada) adalah putih (#fff, baris 12) dan ukuran font yang cukup besar yakni

32 piksel (baris 13). Pointer diubah menjadi tangan saat pengguna mengarahkan kursor (cursor: pointer, baris 14), memberikan indikasi bahwa elemen tersebut bisa diklik.

Untuk tampilan bayangan, box-shadow: 0 2px 10px rgba(0,0,0,0.3) (baris 15) menambahkan efek elevasi lembut agar ikon terlihat mengambang di atas halaman. Properti z-index: 1000 (baris 16) memastikan ikon muncul di atas semua elemen halaman lain. Akhirnya, transition: bottom 0.3s, right 0.3s (baris 17) mengaktifkan transisi lembut saat posisi ikon berubah—misalnya dalam mode mobile.

3. Tampilan Chatbot



```
1  .chat-container-popup {
2    display: none;
3    position: fixed;
4    bottom: 72px;
5    right: 16px;
6    width: 90vw;
7    max-width: 360px;
8    max-height: 80vh;
9    background: #fff;
10   border-radius: 8px;
11   box-shadow: 0 2px 15px rgba(0,0,0,0.3);
12   overflow: hidden;
13   display: flex;
14   flex-direction: column;
15   z-index: 1001;
16 }
```

Gambar 3.4 Tampilan Chatbot

Class .chat-container-popup digunakan untuk mendefinisikan struktur dan tampilan utama jendela chatbot yang muncul setelah ikon diklik. Properti display: none (baris 2) membuat elemen ini tidak terlihat secara default; visibilitasnya dikendalikan melalui JavaScript saat fungsi toggleChat() dipanggil.

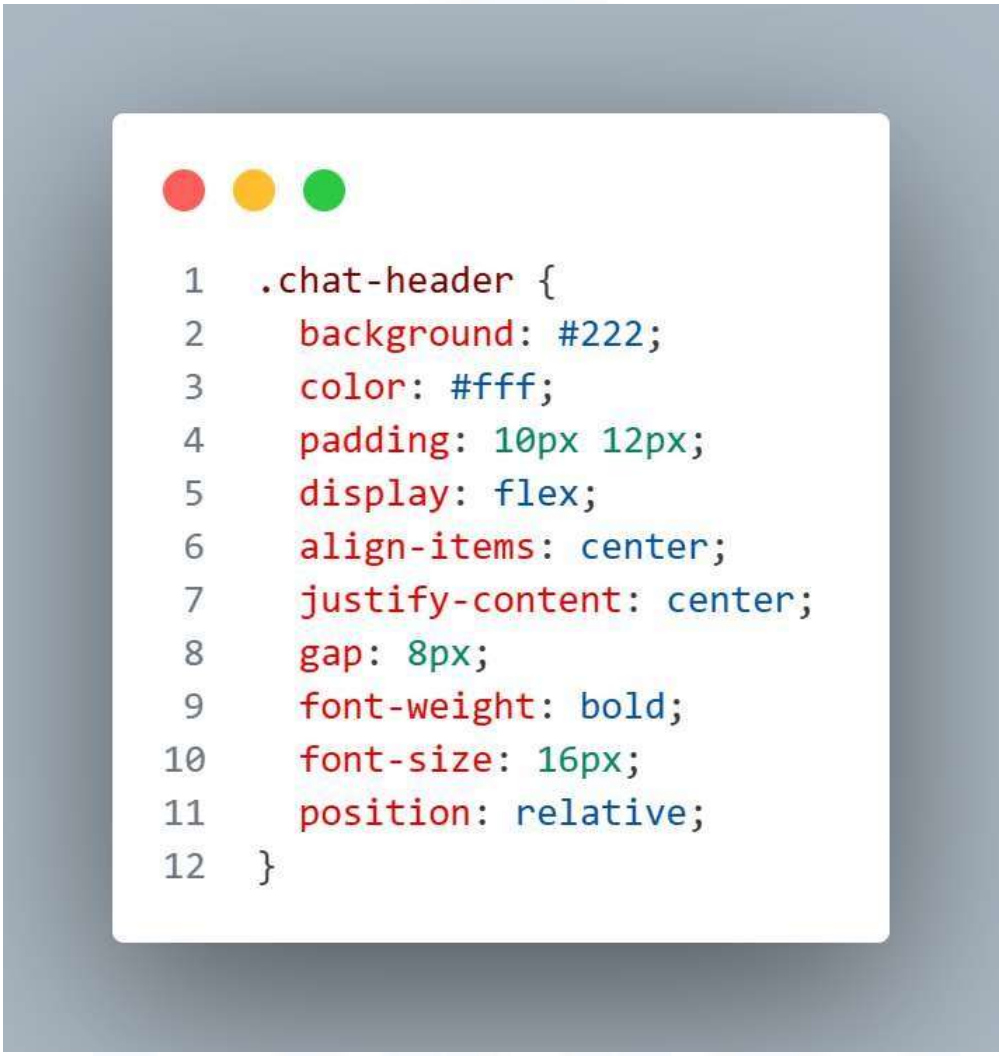
Sama seperti .chat-popup, kontainer ini menggunakan position: fixed (baris 3) agar tetap muncul di posisi yang sama meskipun halaman digulir. Properti bottom: 72px dan right: 16px (baris 4–5) menempatkannya di pojok kanan bawah, sejajar dengan ikon pembuka.

Lebar kontainer diatur dengan width: 90vw (90% dari lebar viewport), tetapi dibatasi maksimum 360px (max-width: 360px, baris 7) agar tetap proporsional. Ketinggian juga dibatasi dengan max-height: 80vh (baris 8), yakni 80% dari tinggi layar pengguna. Ini penting agar chatbot tidak memakan seluruh layar terutama di perangkat kecil.

Latar belakang kontainer dibuat putih (#fff, baris 9), dan sudut-sudutnya dibulatkan sedikit menggunakan border-radius: 8px (baris 10). Efek bayangan diperkuat dengan box-shadow: 0 2px 15px rgba(0,0,0,0.3) (baris 11), membuat jendela terlihat mengambang secara lebih tegas dibandingkan ikon.

Untuk menghindari isi kontainer meluber, digunakan overflow: hidden (baris 12). Kemudian, display: flex dan flex-direction: column (baris 13–14) menata elemen-elemen dalam chatbot agar tersusun secara vertikal (header di atas, pesan di tengah, tombol di bawah). Terakhir, z-index: 1001 (baris 15) memberi prioritas tampilan lebih tinggi daripada ikon (yang menggunakan 1000), memastikan jendela chat tampil di atas ikon.

4. Header Chatbot



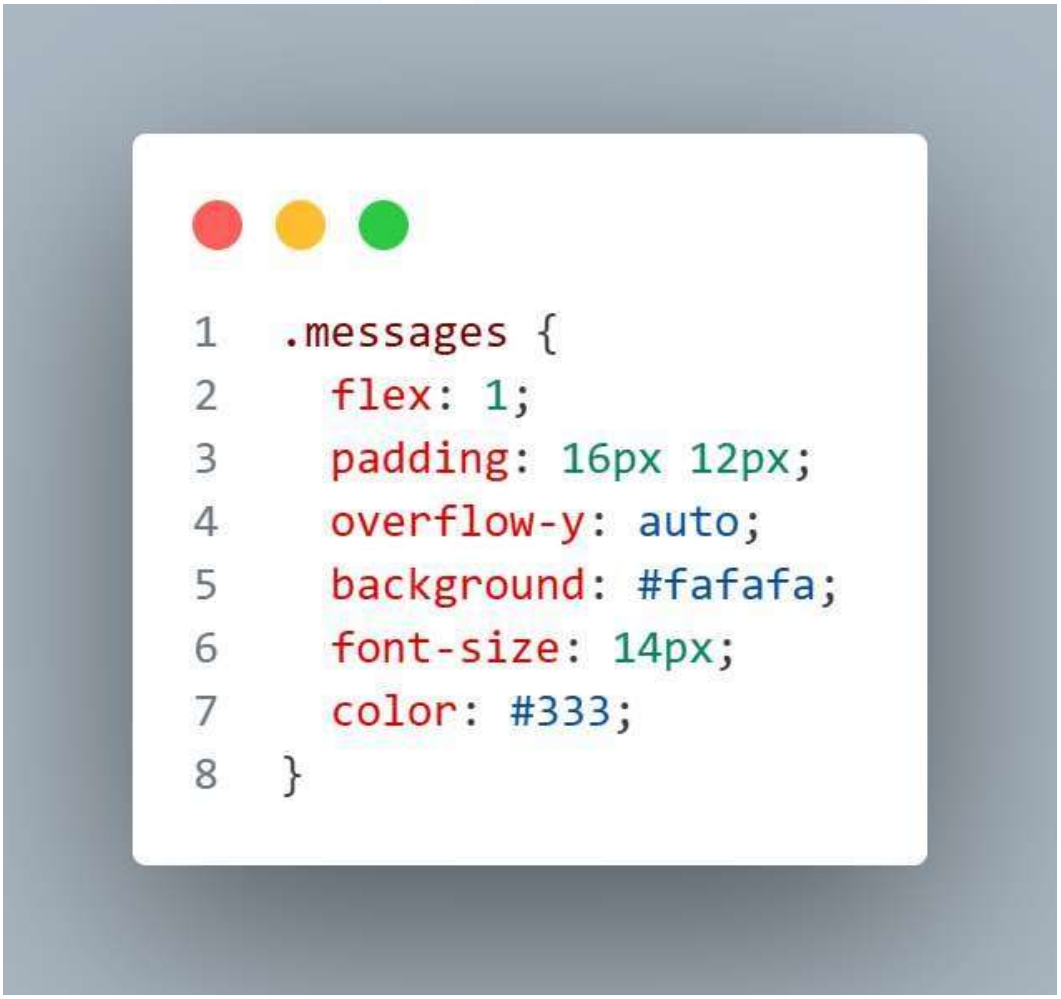
```
1  .chat-header {
2    background: #222;
3    color: #fff;
4    padding: 10px 12px;
5    display: flex;
6    align-items: center;
7    justify-content: center;
8    gap: 8px;
9    font-weight: bold;
10   font-size: 16px;
11   position: relative;
12 }
```

Gambar 3.5 Header Chatbot

Kode ini (baris 1–12) mengatur bagian header dari chatbot. Warna latar belakang hitam pekat (#222, baris 2) dikombinasikan dengan warna teks putih (#fff, baris 3). Padding horizontal dan vertikal diatur agar elemen logo dan teks tidak terlalu rapat (baris 4). display: flex (baris 5) digunakan untuk menyusun logo dan teks dalam satu baris secara horizontal. Elemen disejajarkan secara vertikal (align-items: center, baris 6) dan ditempatkan di

tengah (justify-content: center, baris 7). Jarak antar elemen diatur dengan gap: 8px (baris 8), dan teks “Mitra GMLS” ditampilkan dengan huruf tebal (font-weight: bold, baris 9) serta ukuran 16px (baris 10). Dengan position: relative, tombol tutup (□) bisa diposisikan secara absolut terhadap header ini.

5. Area Messages dan Tampilan Percakapan

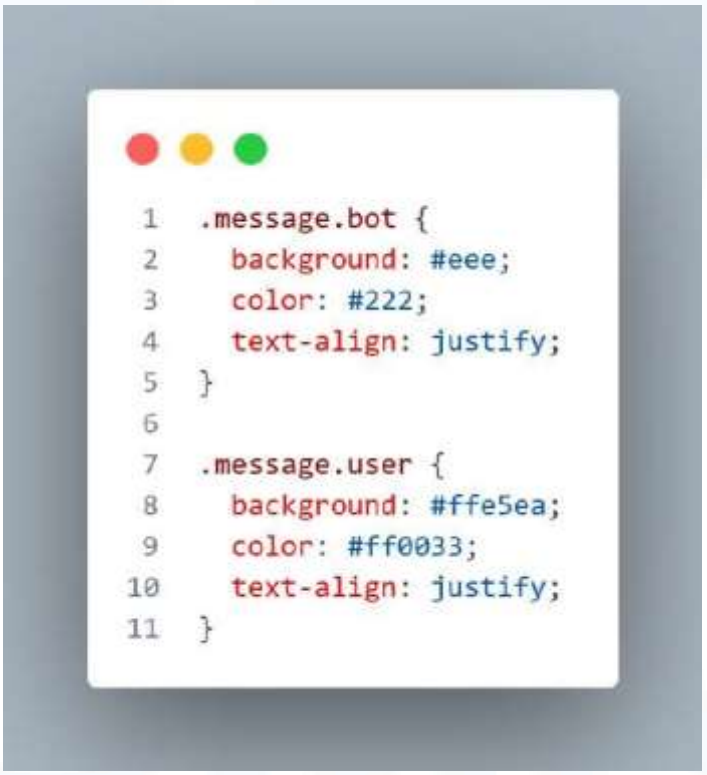


```
1  .messages {
2    flex: 1;
3    padding: 16px 12px;
4    overflow-y: auto;
5    background: #fafafa;
6    font-size: 14px;
7    color: #333;
8  }
```

Gambar 3.6 Area Messages

Class .messages (baris 1–8) merupakan area utama di mana seluruh percakapan ditampilkan. Dengan flex: 1 (baris 2), area ini akan memanjang mengikuti sisa tinggi ruang dari kontainer utama. Padding (16px 12px, baris 3) memastikan konten tidak menempel ke tepi. Jika isi terlalu panjang, scroll vertikal otomatis akan muncul (overflow-y: auto, baris 4). Warna latar belakang abu terang (#fafafa, baris 5) menciptakan kontras halus terhadap balon pesan, dan warna teks abu tua (#333, baris 7) tetap jelas dan enak dibaca.

6. Pesan Bot dan Pengguna



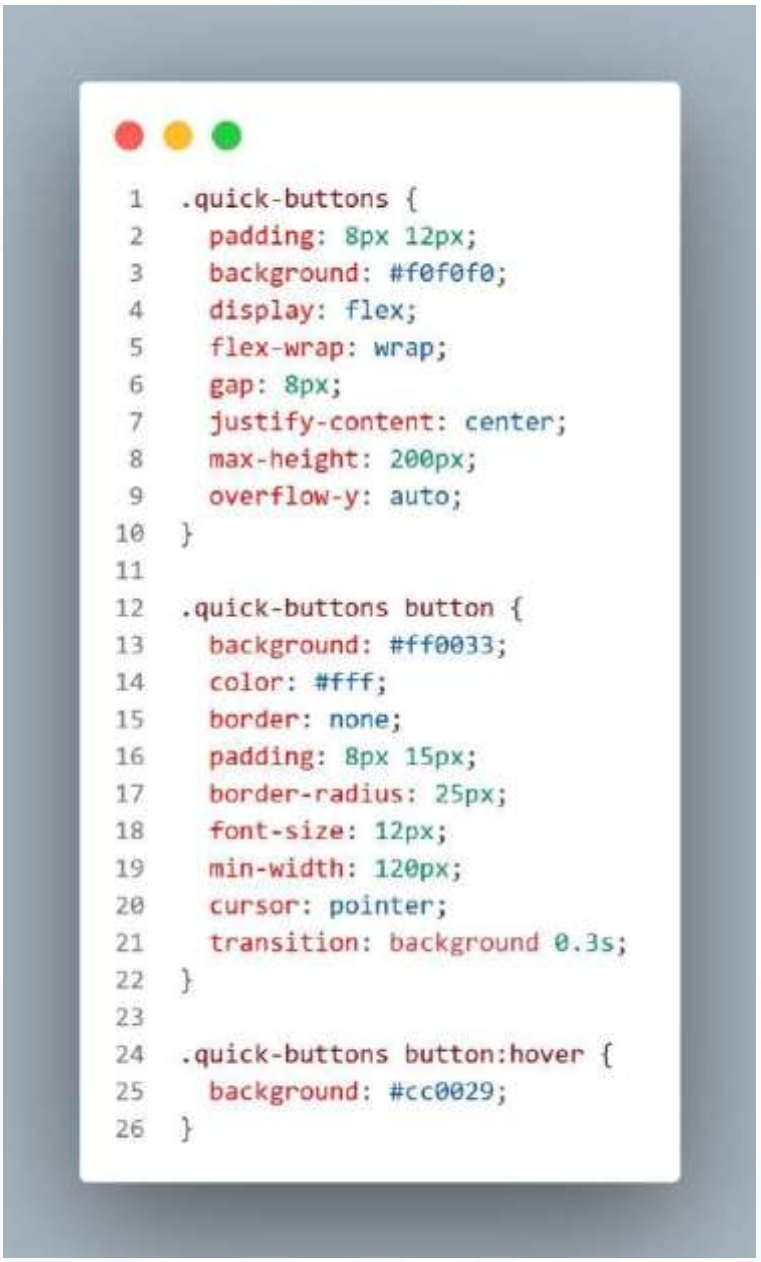
Gambar 3.7 Bot & User Messages

Bagian ini mendefinisikan tampilan visual dari pesan yang dikirim chatbot (.message.bot) dan dari pengguna (.message.user). Pada baris

pertama, .message.bot menetapkan bahwa balon pesan milik chatbot akan memiliki latar abu-abu terang (background: #eee, baris 2), warna teks abu tua (color: #222, baris 3), dan teks yang rata kanan-kiri (text-align: justify, baris 4). Ini memberi kesan informatif dan netral. Sementara itu,

.message.user (baris 7–11) menggunakan latar belakang merah muda lembut (background: #ffe5ea, baris 8) dan teks berwarna merah terang khas GMLS (color: #ff0033, baris 9). Dengan justifikasi teks juga (baris 10), pesan pengguna tetap mudah dibaca dan tampil rapi. Diferensiasi visual ini penting agar pembaca dapat langsung membedakan antara balasan sistem dan pesan yang mereka kirim.

7. Quick Button



```css
.quick-buttons {
  padding: 8px 12px;
  background: #f0f0f0;
  display: flex;
  flex-wrap: wrap;
  gap: 8px;
  justify-content: center;
  max-height: 200px;
  overflow-y: auto;
}

.quick-buttons button {
  background: #ff0033;
  color: #fff;
  border: none;
  padding: 8px 15px;
  border-radius: 25px;
  font-size: 12px;
  min-width: 120px;
  cursor: pointer;
  transition: background 0.3s;
}

.quick-buttons button:hover {
  background: #cc0029;
}
```

Gambar 3.8 Quick Buttons

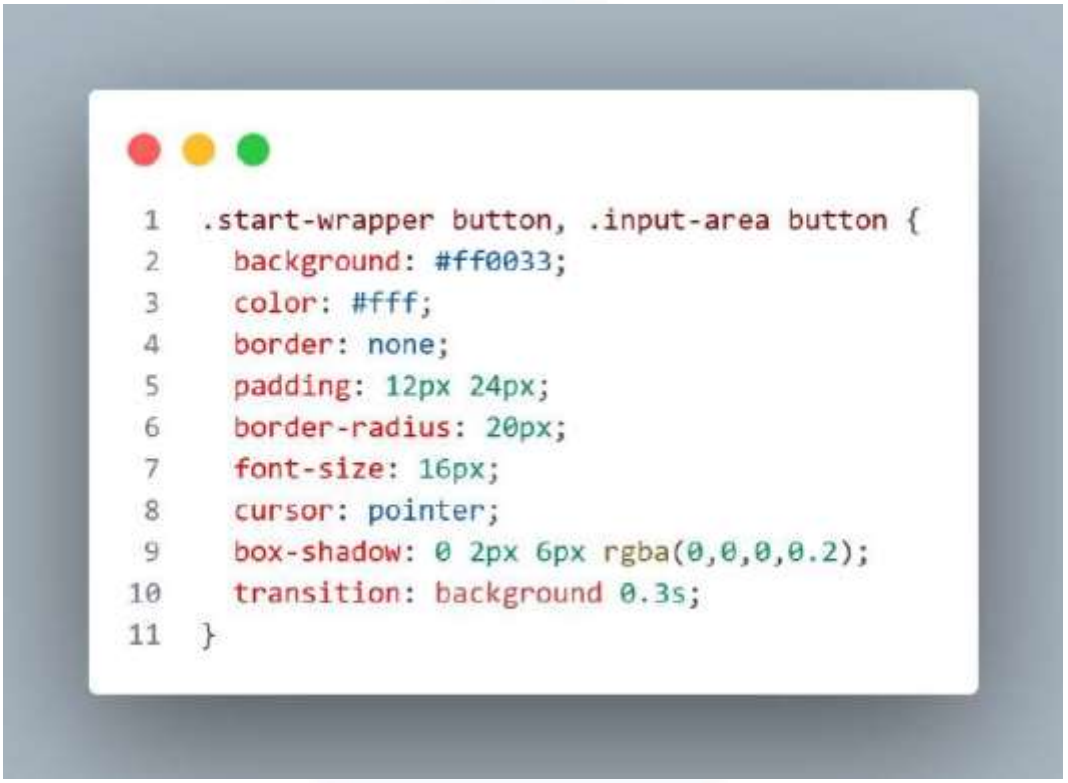
Class .quick-buttons (baris 1–10) digunakan untuk mengatur area tempat tombol-tombol pilihan cepat (quick reply) ditampilkan. Baris 2 mengatur padding di dalam elemen agar tombol tidak mepet ke tepi. Warna latar belakang ditetapkan sebagai abu terang (#f0f0f0, baris 3) agar kontras terhadap tombol merah.

Kemudian, struktur fleksibel dibuat dengan display: flex (baris 4) dan flex-wrap: wrap (baris 5) agar tombol dapat otomatis pindah ke baris bawah jika tidak muat. Jarak antar tombol diatur dengan gap: 8px (baris 6), sedangkan justify-content: center (baris 7) menempatkan tombol-tombol di tengah. Untuk menghindari area ini terlalu panjang, tinggi maksimal ditetapkan sebesar 200px (max-height, baris 8), dan jika melebihi, maka akan muncul scroll vertikal (overflow-y: auto, baris 9).

Gaya tombol-tombol di dalam .quick-buttons ditentukan pada baris 12–22. Warna latar merah mencolok (background: #ff0033, baris 13) dan teks putih (color: #fff, baris 14) memberi kontras yang kuat. Tombol ini tidak memiliki border (border: none, baris 15), dan diberi padding (padding: 8px 15px, baris 16) agar tampak nyaman disentuh.

Bentuknya dibuat lonjong dengan border-radius: 25px (baris 17). Ukuran huruf kecil tetapi jelas (font-size: 12px, baris 18), dan tombol tidak boleh lebih kecil dari 120px lebarnya (min-width: 120px, baris 19). Saat pengguna mengarahkan kursor, pointer berubah (cursor: pointer, baris 20), dan warna tombol berubah secara transisi halus (transition: background 0.3s, baris 21). Warna saat hover berubah menjadi merah lebih gelap (#cc0029, baris 25), menunjukkan interaktivitas tombol.

8. Tombol Mulai dan Restart

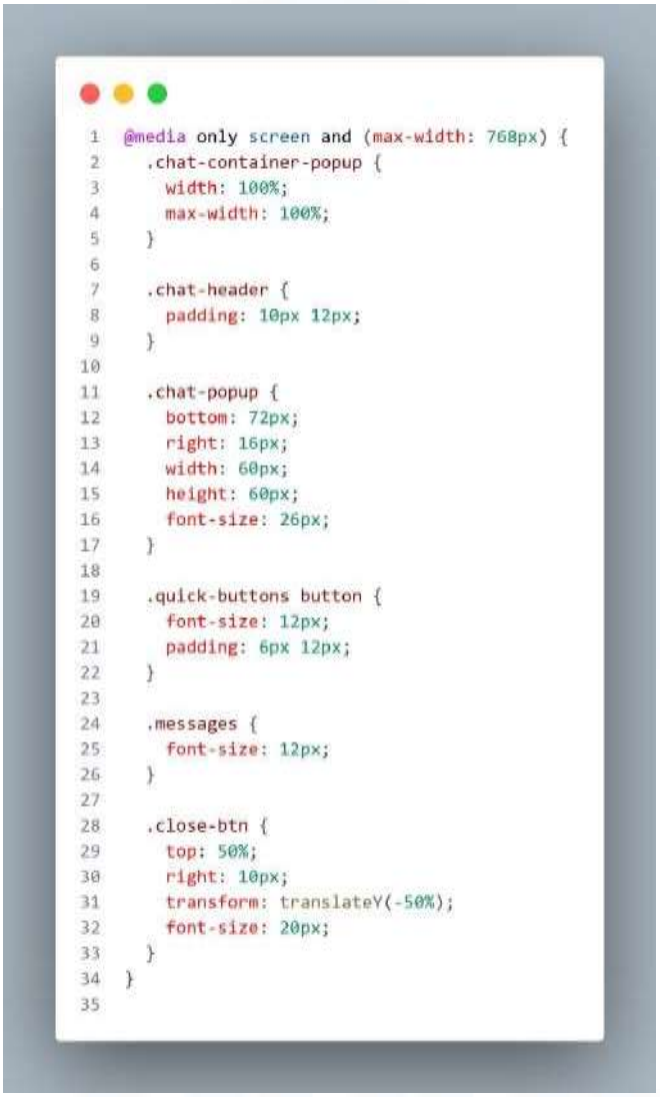


```
1  .start-wrapper button, .input-area button {
2    background: #ff0033;
3    color: #fff;
4    border: none;
5    padding: 12px 24px;
6    border-radius: 20px;
7    font-size: 16px;
8    cursor: pointer;
9    box-shadow: 0 2px 6px rgba(0,0,0,0.2);
10   transition: background 0.3s;
11 }
```

Gambar 3.9 Start & Restart Button

Baris 1–11 ini mengatur tombol besar yang muncul di bagian awal chatbot, seperti tombol “Mulai” dan tombol konfirmasi input. Warna dasar tetap merah GMLS (background: #ff0033, baris 2) dengan teks putih (color: #fff, baris 3). Border dihilangkan (border: none, baris 4) dan tombol diberi padding besar (padding: 12px 24px, baris 5), cocok untuk antarmuka mobile. border-radius: 20px (baris 6) memberikan bentuk oval yang lembut. Font cukup besar (16px, baris 7) agar mudah terlihat. Kursor berubah saat hover (cursor: pointer, baris 8), dan box-shadow (baris 9) memberikan kesan timbul atau melayang. Transisi warna saat hover (baris 10) memberikan animasi halus saat pengguna berinteraksi.

9. Desain Untuk Versi Mobile



```
@media only screen and (max-width: 768px) {
  .chat-container-popup {
    width: 100%;
    max-width: 100%;
  }

  .chat-header {
    padding: 10px 12px;
  }

  .chat-popup {
    bottom: 72px;
    right: 16px;
    width: 60px;
    height: 60px;
    font-size: 26px;
  }

  .quick-buttons button {
    font-size: 12px;
    padding: 6px 12px;
  }

  .messages {
    font-size: 12px;
  }

  .close-btn {
    top: 50%;
    right: 10px;
    transform: translateY(-50%);
    font-size: 20px;
  }
}
```

Gambar 3.10 Desain Mobile Version

Blok ini (baris 1–34) menggunakan media query untuk menyesuaikan tampilan chatbot pada perangkat berlayar kecil seperti ponsel. Kontainer utama diatur agar lebarnya maksimal (width dan max-width) menjadi 100% (baris 2–4), dan padding header disesuaikan agar tidak terlalu besar (baris 7–8). Ukuran ikon chatbot diperkecil (60px

untuk width dan height, baris 13–14), serta font-size ikon diperkecil menjadi 26px (baris 15). Tombol cepat juga dikompres ukurannya (baris 19–21), demikian juga ukuran teks pesan (.messages, baris 24–25). Tombol

juga diperbaiki posisinya agar tetap pas dengan ukuran header yang

kecil (baris 28–33), menggunakan kombinasi top, right, transform, dan font-size.

10. Deklarasi Elemen & Variabel

```
1  const chat = document.getElementById('chat');
2  const quickButtons = document.getElementById('quickButtons');
3  const chatContainer = document.getElementById('chatContainer');
4  const inputArea = document.getElementById('inputArea');
5
6  let lokasiUser = '';
7  let bencanaDipilih = '';
8  let chatEnded = true;
9  let isMinimized = false;
```

Gambar 3.11 Deklarasi Elemen & Variabel

Pada baris 1 hingga 4, digunakan method document.getElementById() untuk mengambil referensi terhadap empat elemen penting dari HTML yang sudah disiapkan sebelumnya. Baris 1 (const chat = document.getElementById('chat');) menyimpan elemen chat ke dalam variabel chat. Elemen ini adalah tempat semua percakapan (pesan bot dan pengguna) akan ditampilkan secara visual dalam bentuk bubble. Baris 2 (const quickButtons = document.getElementById('quickButtons');) mengambil

elemen dengan ID quickButtons, yang berfungsi sebagai tempat tombol-tombol cepat (quick replies) seperti pilihan jenis bencana atau opsi navigasi dalam alur percakapan. Baris 3 (const chatContainer= document.getElementById('chatContainer');) menyimpan elemen container utama yang membungkus keseluruhan jendela chatbot. Elemen ini nantinya akan dikontrol untuk ditampilkan atau disembunyikan berdasarkan interaksi pengguna. Baris 4 (const inputArea = document.getElementById('inputArea');) merujuk ke elemen input pengguna, meskipun dalam implementasi chatbot ini, fungsinya lebih sering digunakan untuk tombol "Mulai" daripada input teks manual.

Selanjutnya pada baris 6 hingga 9, dilakukan deklarasi variabel menggunakan let karena nilainya bersifat dinamis dan akan diubah selama jalannya percakapan. Baris 6 (let lokasiUser = '';) menyimpan lokasi pengguna dalam bentuk string, yang nilainya akan diisi melalui fungsi getLocation() saat chatbot dimulai. Informasi ini nantinya ditampilkan pada awal percakapan sebagai bentuk personalisasi layanan. Baris 7 (let bencanaDipilih = '';) digunakan untuk menyimpan jenis bencana yang sedang dibahas, dan sangat penting karena menjadi parameter dalam pemanggilan fungsi lain seperti getPenjelasan(), getMitigasi(), atau getTempatEvakuasi(). Baris 8 (let chatEnded = true;) menandai status obrolan. Nilai true berarti chatbot belum aktif atau sudah diakhiri, sementara jika sedang berlangsung, nilainya akan diubah menjadi false melalui fungsi startChat(). Terakhir, baris 9 (let isMinimized = false;) merupakan variabel penanda apakah tampilan chatbot sedang diminimalkan atau tidak. Walaupun belum digunakan secara aktif dalam logika yang ada, variabel ini disiapkan sebagai fitur tambahan untuk antarmuka agar nantinya dapat menyimpan status interaksi visual pengguna terhadap tampilan chatbot.

11. Fungsi Closechat

Gambar 3.12 Fungsi Closechat

Fungsi ini digunakan untuk menutup tampilan UI dari chatbot ketika pengguna menekan tombol keluar (biasanya ikon ⎋). Pada baris 2, fungsi mengambil elemen DOM dengan ID chatContainer (kontainer utama chat). Kemudian di baris 3, properti style.display diubah menjadi 'none', yang berarti elemen ini tidak akan ditampilkan di layar, meskipun secara teknis masih ada di dalam DOM. Hal ini memungkinkan pengguna untuk "menyembunyikan" chat tanpa perlu me-refresh halaman atau menghapus isi dari memori. Fungsi ini berguna dalam skenario real-time untuk memberikan kontrol penuh kepada pengguna atas tampilan UI chatbot.

12. Fungsi AppendMessage

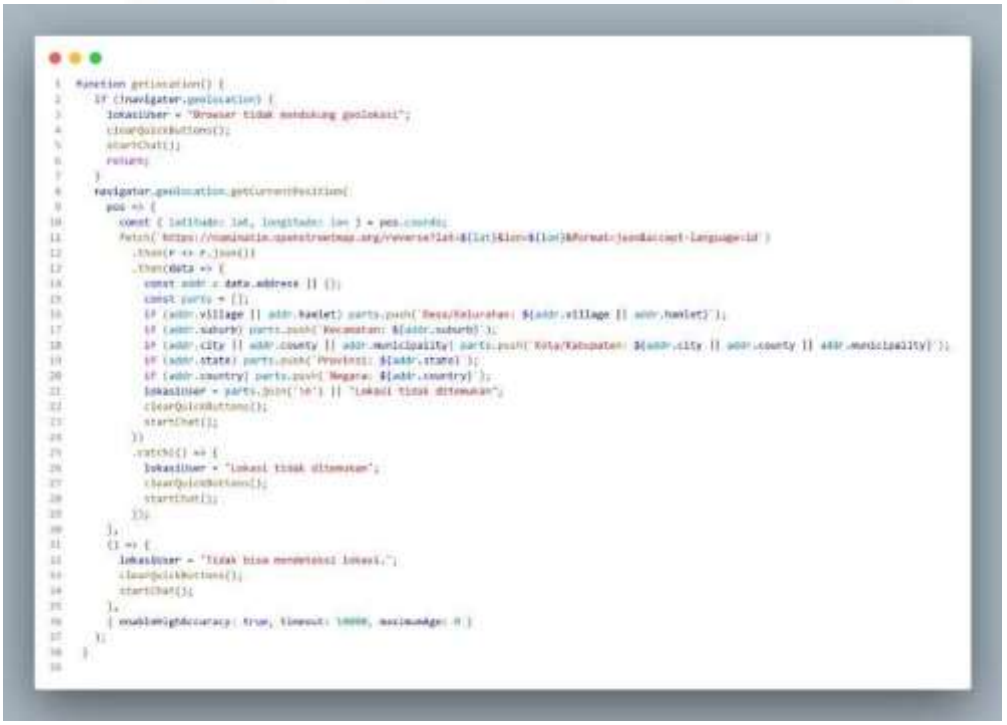
```
1  function appendMessage(text, sender = 'bot') {
2        const wrapper = document.createElement('div');
3        wrapper.className = 'message-with-avatar' + (sender === 'user' ? ' user' : '');
4        const msg = document.createElement('div');
5        msg.className = `message ${sender}`;
6        msg.innerHTML = text.replace(/\n/g, '<br>');
7        const avatar = document.createElement('img');
8        avatar.src = sender === 'bot'
9          ? 'https://gmls.org/wp-content/uploads/2025/06/Avatar_GMLS.png'
10         : 'https://gmls.org/wp-content/uploads/2025/06/Avatar_User.png';
11       avatar.alt = sender === 'bot' ? 'Bot' : 'User';
12
13       if (sender === 'bot') {
14         wrapper.append(avatar, msg);
15       } else {
16         wrapper.append(msg, avatar);
17       }
18
19       chat.appendChild(wrapper);
20       chat.scrollTop = chat.scrollHeight;
21     }
```

Gambar 3.13 Fungsi AppendMessage

Fungsi appendMessage adalah fungsi utama yang digunakan untuk menampilkan setiap pesan di antarmuka percakapan chatbot. Fungsi ini menerima dua parameter: text (isi pesan) dan sender (pengirim pesan), dengan nilai default 'bot'. Di baris 2, dibuat elemen bernama wrapper sebagai pembungkus dari setiap pesan dan avatarnya. Kemudian pada baris 3, elemen ini diberi class message-with-avatar, dan jika pengirimnya adalah 'user', maka class tambahan 'user' akan ditambahkan agar bisa dibedakan dalam styling. Pada baris 4, elemen baru untuk teks pesan disiapkan dengan nama msg, dan di baris 5 diberi class dinamis sesuai siapa pengirimnya (message bot atau message user). Baris 6 mengatur isi pesan, sekaligus mengganti karakter newline \n menjadi
 agar tetap tampil dengan baris baru di HTML. Di baris 7–11, fungsi membuat elemen gambar avatar dan menetapkan sumber gambar yang sesuai—avatar GMLS

jika pengirimnya adalah bot, dan avatar pengguna jika pengirimnya adalah user. Atribut alt juga ditentukan untuk aksesibilitas dan kejelasan elemen gambar. Kemudian pada baris 13–17, fungsi menentukan posisi elemen avatar dan pesan (msg) dalam wrapper, tergantung dari siapa pengirimnya. Jika pengirim adalah bot, maka avatar tampil di kiri dan pesan di kanan; sebaliknya untuk pengguna. Baris 19 menambahkan wrapper ke dalam elemen chat, dan baris 20 memastikan scroll otomatis bergeser ke bawah agar pesan terbaru langsung terlihat. Fungsi ini memastikan visualisasi percakapan terlihat konsisten dan responsif.

13. Fungsi Geolokasi



Gambar 3.14 Fungsi Geolokasi

Fungsi getLocation() merupakan komponen kunci dalam chatbot Mitra GMLS yang bertanggung jawab untuk memperoleh lokasi geografis pengguna secara otomatis menggunakan fitur Geolocation API dari browser. Proses dimulai dengan pengecekan apakah browser mendukung fitur navigator.geolocation. Bila tidak didukung (baris 2), maka variabel lokasiUser akan diisi dengan pesan "Browser tidak mendukung geolokasi", kemudian tombol-tombol cepat (quick buttons) dibersihkan dengan clearQuickButtons(), dan chatbot tetap melanjutkan ke tahap startChat() meskipun tanpa informasi lokasi (baris 3–5).

Namun, jika geolokasi tersedia (baris 6), maka fungsi getCurrentPosition() dijalankan untuk mencoba mengambil koordinat pengguna. Jika berhasil (baris 7–30), koordinat lintang dan bujur (latitude dan longitude) diekstrak dari objek pos.coords. Selanjutnya, chatbot mengirimkan request menggunakan API reverse geocoding dari Nominatim (OpenStreetMap) (baris 9) untuk menerjemahkan koordinat tersebut menjadi informasi lokasi yang lebih manusiawi, seperti desa, kecamatan, kabupaten, hingga negara. Hasil JSON dari API ini ditangkap dan diolah (baris 11–21), lalu dipetakan ke array parts[] sesuai struktur alamat: village/hamlet, suburb, city/county/municipality, state, dan country (baris 14–18). Setiap bagian yang ditemukan akan dimasukkan ke dalam parts[] dengan format label seperti “Desa/Kelurahan: ...”, hingga akhirnya seluruh bagian disatukan menjadi satu string melalui parts.join('\n'), dan disimpan kembali ke dalam lokasiUser.

Jika terjadi kegagalan dalam proses permintaan API (misalnya error saat fetch atau koneksi), maka blok .catch() akan dijalankan (baris 22–26), dan lokasiUser akan diberi nilai fallback "Lokasi tidak ditemukan".

Bahkan, bila proses pengambilan posisi awal (GPS) gagal total, misalnya karena pengguna menolak permintaan izin lokasi, maka fallback tambahan juga disediakan (baris 31–34) dengan pesan "Tidak bisa mendeteksi lokasi.". Dalam semua skenario, chatbot akan tetap lanjut memulai chat dengan memanggil startChat() agar alur tetap berjalan lancar, meskipun dengan atau tanpa lokasi pengguna. Penggunaan parameter { enableHighAccuracy: true, timeout: 10000, maximumAge: 0 } (baris 35) bertujuan untuk mengoptimalkan akurasi lokasi dengan mengizinkan GPS (bila tersedia), menetapkan batas waktu maksimal 10 detik untuk mendapatkan lokasi, dan memastikan tidak ada data lokasi lama yang dipakai ulang. Secara keseluruhan, fungsi ini menunjukkan pendekatan user-centric yang cerdas, dengan memberikan fallback dan alur tanggapan yang mulus jika terjadi masalah, serta mendukung konteks chatbot edukatif yang berbasis lokasi agar pengguna mendapatkan informasi kebencanaan yang lebih relevan.

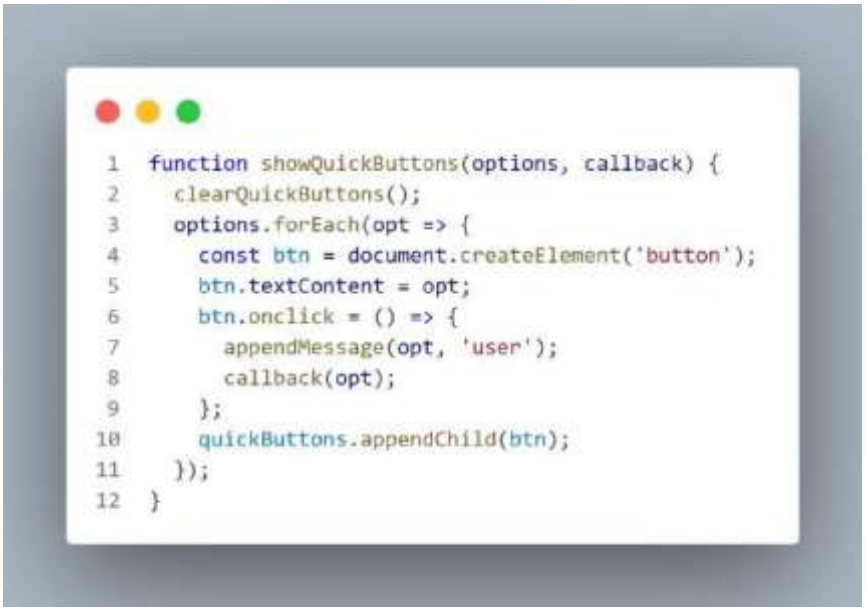
14. Fungsi Startchat

Gambar 3.15 Fungsi Startchat

Fungsi ini merupakan titik awal percakapan antara chatbot dan pengguna. Pada baris 2, chatEnded diatur ke false yang artinya sesi obrolan

sedang berlangsung. Kemudian di baris 3, fungsi appendBotIntroImage() dipanggil untuk menampilkan avatar bot GMLS di awal percakapan—biasanya sebagai elemen pembuka yang menunjukkan kehadiran bot secara visual. Baris 4 menambahkan pesan sapaan awal dari bot dengan teks yang menyebutkan nama bot ("Tyo") dan menjelaskan bahwa ini adalah chatbot edukasi bencana. Dalam teks sapaan ini juga dicantumkan informasi lokasi pengguna yang sudah disimpan di lokasiUser sebelumnya. Penggunaan \n bertujuan untuk memberikan pemisah antarbaris agar teks lebih mudah dibaca. Setelah sapaan, baris 5 menjalankan showBencanaMenu(), yang akan memunculkan pilihan jenis bencana kepada pengguna. Terakhir, baris 6 menghapus seluruh konten dari area input agar tampilan tetap bersih saat sesi baru dimulai.

15. Fungsi Show Quick Buttons



Gambar 3.16 Fungsi ShowQuickButtons

Fungsi showQuickButtons berperan penting dalam menampilkan pilihan interaktif kepada pengguna dalam bentuk tombol cepat (quick buttons). Fungsi ini menerima dua parameter utama, yaitu options yang merupakan array dari pilihan-pilihan yang ingin ditampilkan sebagai tombol, dan callback, yaitu fungsi yang akan dipanggil saat pengguna memilih salah satu tombol. Langkah pertama dalam fungsi ini adalah memanggil clearQuickButtons(), yaitu fungsi yang akan menghapus semua tombol yang sebelumnya ada, agar tombol-tombol baru dapat ditampilkan tanpa terjadi penumpukan.

Selanjutnya, fungsi ini menggunakan forEach untuk mengiterasi setiap elemen dalam array options. Untuk setiap elemen opt, fungsi akan membuat elemen HTML <button> baru menggunakan document.createElement('button'). Isi teks tombol tersebut diatur dengan properti textContent sesuai nilai dari opt, sehingga tombol tersebut akan menampilkan label yang informatif bagi pengguna. Lalu, ditetapkan sebuah event handler onclick untuk tombol tersebut, di mana saat tombol diklik, chatbot akan menjalankan dua hal sekaligus: pertama, menampilkan kembali pesan yang sama seperti yang diklik oleh pengguna dengan appendMessage(opt, 'user'); dan kedua, memicu callback(opt) untuk melanjutkan logika percakapan berdasarkan pilihan tersebut.

Setelah event handler selesai dibuat, tombol tersebut ditambahkan ke elemen induk quickButtons menggunakan appendChild, sehingga akan muncul di antarmuka pengguna. Dengan pendekatan ini, chatbot menjadi lebih interaktif dan mudah digunakan karena pengguna tidak perlu mengetik secara manual, cukup menekan tombol yang sesuai. Fungsi ini dirancang modular dan fleksibel, sehingga bisa digunakan dalam berbagai

konteks pertanyaan atau percabangan dialog dalam chatbot edukatif mitigasi bencana seperti Mitra GMLS.

16. Fungsi Show Info

```
1  function showInfoMenu() {
2    const info = [
3      `Apa itu ${bencanaDipilih}?`,
4      ...(bencanaDipilih === "Tsunami" ? ["Indikator Tsunami Ready"] : []),
5      `Cara mitigasi ${bencanaDipilih}`,
6      `Kit darurat ${bencanaDipilih}`,
7      `Tempat evakuasi ${bencanaDipilih}`,
8      "Chat Support Dengan Tim Kami",
9      "Kembali ke menu bencana",
10     "Akhiri chat"
11   ];
12   appendMessage("Pilih informasi:");
13   showQuickButtons(info, handleInfoChoice);
14 }
```

Gambar 3.17 Fungsi ShowInfoMenu

Fungsi ini digunakan untuk menampilkan daftar informasi spesifik dari bencana yang telah dipilih. Pada baris 2–10, didefinisikan array info yang menyesuaikan dengan bencanaDipilih. Jika bencanaDipilih === "Tsunami" maka akan muncul informasi tambahan yaitu “Indikator Tsunami Ready” (baris 4). Sisanya adalah pilihan standar: penjelasan bencana, cara mitigasi, kit darurat, tempat evakuasi, serta opsi bantuan chat, kembali ke menu bencana, atau mengakhiri chat. Baris 12 mengirimkan pesan "Pilih informasi:", dan baris 13 memanggil showQuickButtons() dengan parameter info dan handleInfoChoice.

17. Fungsi Back Options

```
1  function backOptions() {
2    showQuickButtons(
3      ["Kembali ke info", "Kembali ke bencana", "Akhiri chat"],
4      choice => {
5        if (choice === "Kembali ke info") {
6          showInfoMenu();
7        } else if (choice === "Kembali ke bencana") {
8          showBencanaMenu();
9        } else {
10         appendMessage("✅ Terima kasih telah menggunakan Mitra GMLS.", "bot");
11         chatEnded = true;
12         clearQuickButtons();
13         showRestartInMiddle();
14       }
15     }
16   );
17 }
```

Gambar 3.18 Fungsi BackOptions

Fungsi ini (baris 1) digunakan untuk menampilkan pilihan ketika pengguna ingin kembali setelah mendapatkan informasi tentang bencana. Baris 2 memanggil showQuickButtons() dengan tiga opsi: "Kembali ke info", "Kembali ke bencana", dan "Akhiri chat". Pada baris 4–15, terdapat fungsi callback choice => { ... } yang menentukan aksi berdasarkan pilihan pengguna. Jika pengguna memilih "Kembali ke info", maka dipanggil fungsi

showInfoMenu() (baris 6). Jika pengguna memilih "Kembali ke bencana", maka chatbot akan kembali ke daftar bencana dengan memanggil

showBencanaMenu() (baris 8). Namun jika memilih "Akhiri chat", maka chatbot akan mengirimkan pesan penutup (baris 10), mengubah status chatEnded menjadi true, membersihkan tombol cepat (baris 12), dan menampilkan ulang tombol "Mulai" di tengah (baris 13).

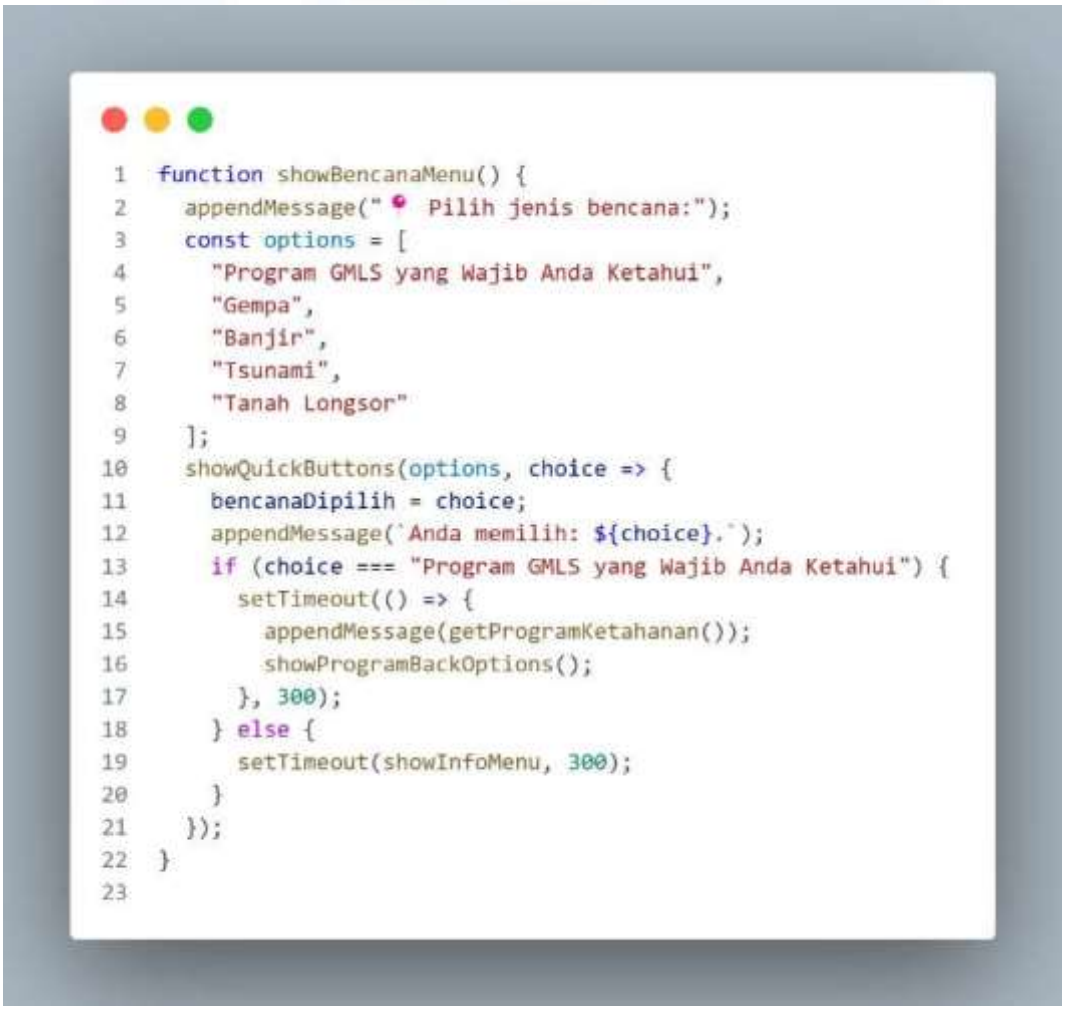
18. Fungsi Show Program Back Option

```
1  function showProgramBackOptions() {
2    showQuickButtons(
3      ["Kembali ke bencana", "Akhiri chat"],
4      choice => {
5        if (choice === "Kembali ke bencana") {
6          showBencanaMenu();
7        } else {
8          appendMessage("✅ Terima kasih telah menggunakan Mitra GMLS.", "bot");
9          chatEnded = true;
10         clearQuickButtons();
11         showRestartInMiddle();
12       }
13     }
14   );
15 }
16
```

Gambar 3.19 Fungsi ShowProgramBackOption

Fungsi ini mirip dengan backOptions() namun khusus digunakan ketika pengguna memilih opsi “Program GMLS yang Wajib Anda Ketahui”. Pada baris 2 dipanggil showQuickButtons() dengan dua opsi saja: "Kembali ke bencana" dan "Akhiri chat". Baris 4–12 berisi callback yang memeriksa apakah pengguna ingin kembali ke menu bencana (baris 5–6), atau jika tidak, maka chat akan diakhiri (baris 8–11) sama seperti sebelumnya: menampilkan pesan, mengubah status, membersihkan tombol cepat, dan memunculkan tombol restart.

19. Fungsi Show Bencana Menu

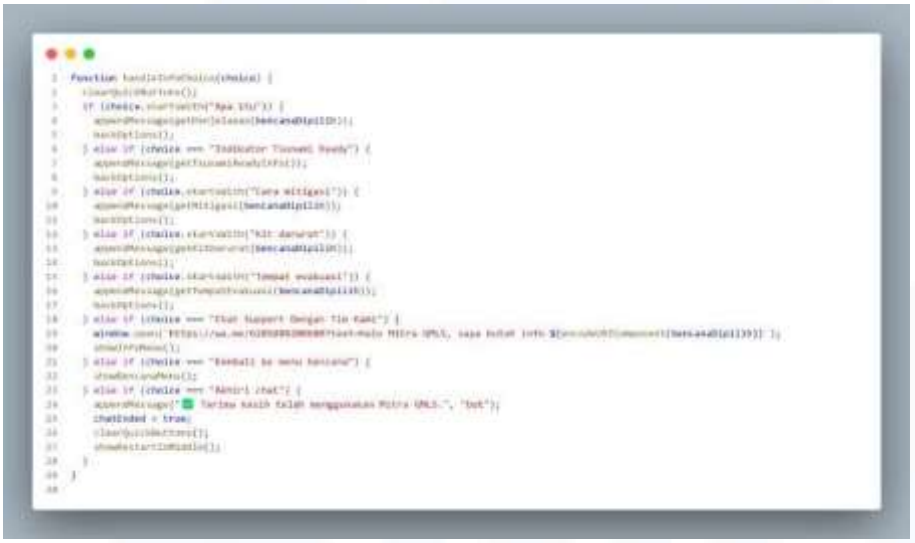


```
function showBencanaMenu() {
  appendMessage("📍 Pilih jenis bencana:");
  const options = [
    "Program GMLS yang Wajib Anda Ketahui",
    "Gempa",
    "Banjir",
    "Tsunami",
    "Tanah Longsor"
  ];
  showQuickButtons(options, choice => {
    bencanaDipilih = choice;
    appendMessage(`Anda memilih: ${choice}.`);
    if (choice === "Program GMLS yang Wajib Anda Ketahui") {
      setTimeout(() => {
        appendMessage(getProgramKetahanan());
        showProgramBackOptions();
      }, 300);
    } else {
      setTimeout(showInfoMenu, 300);
    }
  });
}
```

Gambar 3.20 Fungsi ShowBencana

Fungsi ini adalah salah satu fungsi utama yang menampilkan daftar jenis bencana. Pada baris 2, chatbot menyapa dengan pesan untuk memilih jenis bencana. Baris 3–9 mendefinisikan daftar opsi berupa array options, seperti "Program GMLS yang Wajib Anda Ketahui", "Gempa", "Banjir", "Tsunami", dan "Tanah Longsor". Baris 10 memanggil showQuickButtons() dan menambahkan fungsi callback untuk menyimpan pilihan ke dalam variabel bencanaDipilih (baris 11), lalu menampilkan pesan konfirmasi pilihan pengguna (baris 12). Jika yang dipilih adalah “Program GMLS”, maka (baris 13–17) akan dijalankan appendMessage() untuk menampilkan isi program, lalu memunculkan tombol back khusus program. Jika bukan, maka dipanggil showInfoMenu() dengan penundaan 300 milidetik (baris 19).

20. Fungsi Handle Info Choice



Gambar 3.21 Fungsi HandleInfoChoice

Fungsi handleInfoChoice(choice) (baris 1) adalah salah satu fungsi utama dalam sistem percakapan chatbot ini, yang bertugas menangani

berbagai pilihan jawaban yang diberikan pengguna ketika ditampilkan menu informasi seputar bencana. Fungsi ini menerima satu parameter choice, yang merepresentasikan teks dari tombol yang dipilih oleh pengguna. Pada baris 2, fungsi langsung membersihkan tampilan tombol cepat sebelumnya dengan memanggil clearQuickButtons(), agar tombol-tombol baru bisa ditampilkan tanpa tumpang tindih.

Selanjutnya pada baris 3 hingga 5, terdapat logika pertama: jika choice dimulai dengan kata "Apa itu" (menggunakan metode .startsWith()), maka fungsi akan memanggil getPenjelasan(bencanaDipilih) untuk menampilkan deskripsi tentang jenis bencana yang dipilih, lalu memunculkan kembali tombol navigasi backOptions() agar pengguna dapat kembali memilih menu sebelumnya.

Kemudian, di baris 6 hingga 8, jika choice adalah "Indikator Tsunami Ready" (hanya berlaku untuk jenis bencana tsunami), maka fungsi akan menjalankan getTsunamiReadyInfo() yang berisi informasi mengenai kesiapsiagaan tsunami berbasis komunitas, lalu juga menampilkan backOptions(). Baris 9–11 memeriksa apakah choice dimulai dengan "Cara mitigasi", dan bila benar, akan menjalankan getMitigasi(bencanaDipilih) untuk memberikan panduan mitigasi terhadap bencana yang dipilih, kemudian kembali menampilkan tombol navigasi backOptions(). Pola ini diulang juga untuk opsi "Kit darurat" (baris 12–14) dan "Tempat evakuasi" (baris 15–17), di mana setiap pilihan akan menjalankan fungsi masing-masing: getKitDarurat() dan getTempatEvakuasi() yang keduanya merujuk pada nilai dari variabel bencanaDipilih.

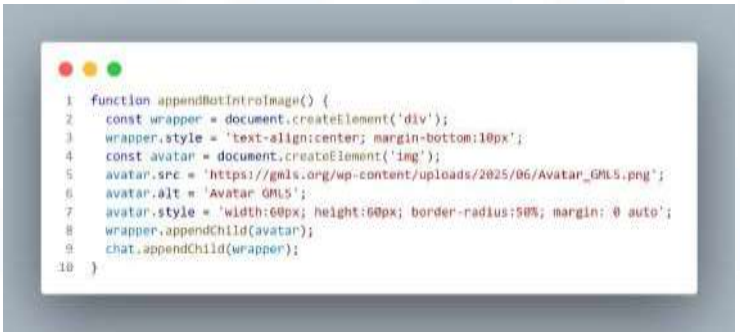
Salah satu fitur penting lainnya ada di baris 18–20, yaitu saat pengguna memilih “Chat Support Dengan Tim Kami”. Dalam kondisi ini,

fungsi akan membuka tab baru ke WhatsApp menggunakan window.open() dan membentuk pesan otomatis berisi permintaan informasi yang disesuaikan dengan jenis bencana yang dipilih. Parameter bencana akan dienkode melalui encodeURIComponent(bencanaDipilih) agar tidak rusak saat dimasukkan ke dalam URL. Setelah itu, menu informasi akan ditampilkan kembali lewat showInfoMenu().

Jika pengguna memilih "Kembali ke menu bencana" (baris 21–22), maka fungsi akan menampilkan ulang daftar jenis bencana melalui showBencanaMenu().

Terakhir, apabila pengguna memilih "Akhiri chat" (baris 23–27), maka sistem menampilkan pesan penutup berupa ucapan terima kasih menggunakan appendMessage(...), mengatur flag chatEnded menjadi true agar bot tahu percakapan telah berakhir, menghapus tombol-tombol cepat dengan clearQuickButtons(), dan memunculkan tombol "Mulai" kembali di tengah layar dengan memanggil showRestartInMiddle().

21. Fungsi Append Bot Intro Image



```
1  function appendBotIntroImage() {
2    const wrapper = document.createElement('div');
3    wrapper.style = 'text-align:center; margin-bottom:10px';
4    const avatar = document.createElement('img');
5    avatar.src = 'https://gmls.org/wp-content/uploads/2025/06/Avatar_GMLS.png';
6    avatar.alt = 'Avatar GMLS';
7    avatar.style = 'width:60px; height:60px; border-radius:50%; margin: 0 auto';
8    wrapper.appendChild(avatar);
9    chat.appendChild(wrapper);
10 }
```

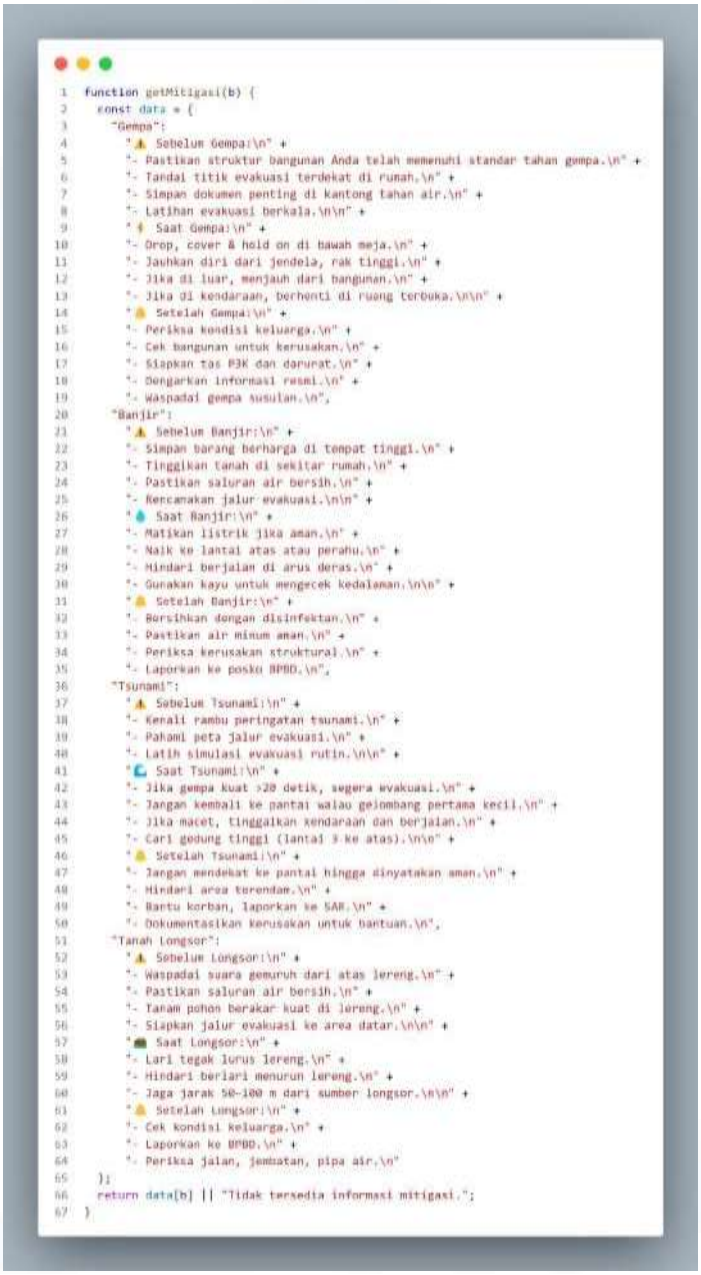
Gambar 3.22 Fungsi AppendBotIntroImage

Pada baris 1, fungsi appendBotIntroImage() didefinisikan tanpa parameter karena tujuannya hanya untuk menambahkan elemen visual ke

dalam tampilan chatbot. Kemudian di baris 2, dibuat elemen div baru bernama wrapper yang akan menjadi kontainer dari gambar bot. Di baris 3, style dari wrapper diatur secara langsung menggunakan properti textAlign

= "center" agar isi kontainer diratakan ke tengah, dan marginBottom = "10px" untuk memberi jarak antar elemen secara vertikal. Selanjutnya pada baris 4, dibuat elemen img bernama avatar sebagai gambar bot. Di baris 5, atribut src dari gambar diisi dengan URL logo GMLS. Kemudian di baris 6, alt text diberikan sebagai deskripsi jika gambar gagal dimuat. Di baris 7, gaya visual gambar ditambahkan secara langsung: lebar dan tinggi 60 piksel agar proporsional, borderRadius 50% supaya membentuk lingkaran, dan margin = "0 auto" untuk memusatkan secara horizontal. Gambar avatar lalu dimasukkan ke dalam wrapper di baris 8 menggunakan appendChild(), dan terakhir baris 9 menyisipkan wrapper ke dalam elemen chat, sehingga avatar bot ditampilkan secara permanen di awal percakapan.

22. Fungsi Penjelasan Mitigasi Bencana



```
function getMitigasi(b) {
  const data = {
    "Gempa":
      "⚠️ Sebelum Gempa:\n" +
      "- Pastikan struktur bangunan Anda telah memenuhi standar tahan gempa.\n" +
      "- Tandai titik evakuasi terdekat di rumah.\n" +
      "- Simpan dokumen penting di kantong tahan air.\n" +
      "- Latihan evakuasi berkala.\n\n" +
      "🌍 Saat Gempa:\n" +
      "- Drop, cover & hold on di bawah meja.\n" +
      "- Jauhkan diri dari jendela, rak tinggi.\n" +
      "- Jika di luar, menjauh dari bangunan.\n" +
      "- Jika di kendaraan, berhenti di ruang terbuka.\n\n" +
      "🔔 Setelah Gempa:\n" +
      "- Periksa kondisi keluarga.\n" +
      "- Cek bangunan untuk kerusakan.\n" +
      "- Siapkan tas P3K dan darurat.\n" +
      "- Dengarkan informasi resmi.\n" +
      "- Waspadai gempa susulan.\n",
    "Banjir":
      "⚠️ Sebelum Banjir:\n" +
      "- Simpan barang berharga di tempat tinggi.\n" +
      "- Tinggikan tanah di sekitar rumah.\n" +
      "- Pastikan saluran air bersih.\n" +
      "- Rencanakan jalur evakuasi.\n\n" +
      "💧 Saat Banjir:\n" +
      "- Matikan listrik jika aman.\n" +
      "- Naik ke lantai atas atau perahu.\n" +
      "- Hindari berjalan di arus deras.\n" +
      "- Gunakan kayu untuk mengecek kedalaman.\n\n" +
      "🔔 Setelah Banjir:\n" +
      "- Bersihkan dengan disinfektan.\n" +
      "- Pastikan air minum aman.\n" +
      "- Periksa kerusakan struktural.\n" +
      "- Laporkan ke posko BPBD.\n",
    "Tsunami":
      "⚠️ Sebelum Tsunami:\n" +
      "- Kenali rambu peringatan tsunami.\n" +
      "- Pahami peta jalur evakuasi.\n" +
      "- Latih simulasi evakuasi rutin.\n\n" +
      "🌊 Saat Tsunami:\n" +
      "- Jika gempa kuat >20 detik, segera evakuasi.\n" +
      "- Jangan kembali ke pantai walau gelombang pertama kecil.\n" +
      "- Jika macet, tinggalkan kendaraan dan berjalan.\n" +
      "- Cari gedung tinggi (lantai 3 ke atas).\n\n" +
      "🔔 Setelah Tsunami:\n" +
      "- Jangan mendekat ke pantai hingga dinyatakan aman.\n" +
      "- Hindari area terendam.\n" +
      "- Bantu korban, laporkan ke SAR.\n" +
      "- Dokumentasikan kerusakan untuk bantuan.\n",
    "Tanah Longsor":
      "⚠️ Sebelum Longsor:\n" +
      "- Waspadai suara gemuruh dari atas lereng.\n" +
      "- Pastikan saluran air bersih.\n" +
      "- Tanam pohon berakar kuat di lereng.\n" +
      "- Siapkan jalur evakuasi ke area datar.\n\n" +
      "🏃 Saat Longsor:\n" +
      "- Lari tegak lurus lereng.\n" +
      "- Hindari berlari menurun lereng.\n" +
      "- Jaga jarak 50–100 m dari sumber longsor.\n\n" +
      "🔔 Setelah Longsor:\n" +
      "- Cek kondisi keluarga.\n" +
      "- Laporkan ke BPBD.\n" +
      "- Periksa jalan, jembatan, pipa air.\n"
  };
  return data[b] || "Tidak tersedia informasi mitigasi.";
}
```

Gambar 3.23 Fungsi Penjelasan Mitigasi Bencana

23. Fungsi Penjelasan Kit Darurat Bencana

Gambar 3.24 Fungsi Penjelasan Kit Darurat

3.3 Kendala

Dalam pelaksanaan proyek kemanusiaan ini, terdapat beberapa kendala yang dihadapi baik dari sisi teknis maupun non-teknis. Kendala tersebut antara lain:

1. Keterbatasan Waktu Komunikasi dengan Mitra GMLS

Salah satu tantangan utama yang dihadapi adalah koordinasi dengan tim Gugus Mitigasi Lebak Selatan (GMLS) sebagai mitra lapangan. GMLS merupakan organisasi komunitas yang memiliki tanggung jawab besar dalam kegiatan lapangan kebencanaan. Hal ini menyebabkan jadwal mereka cukup padat dan tidak selalu sinkron dengan jadwal pelaksanaan proyek. Akibatnya, terdapat keterlambatan dalam pelaksanaan.

2. Terbatasnya Akses Internet di Wilayah Lapangan

Wilayah Kecamatan Panggarangan sebagai lokasi pelaksanaan proyek termasuk dalam daerah pesisir yang tidak seluruhnya terjangkau oleh sinyal internet yang stabil. Kondisi ini menjadi hambatan ketika melakukan demonstrasi chatbot secara langsung kepada mitra atau masyarakat, karena aplikasi berbasis web memerlukan koneksi internet untuk memuat data dan fungsi chatbot secara optimal.

3. Penyesuaian Desain dengan Website Mitra

Selama masa pengembangan chatbot, pihak GMLS melakukan pembaruan terhadap tampilan dan struktur website resmi mereka. Perubahan ini berdampak langsung pada proses integrasi chatbot ke dalam website karena struktur kode dan estetika visual yang telah disesuaikan sebelumnya menjadi tidak relevan. Perlu melakukan penyesuaian ulang baik dari sisi desain agar tampilan chatbot tetap serasi dengan antarmuka baru yang digunakan oleh GMLS.

3.4 Solusi Dari Kendala

Menghadapi kendala-kendala yang muncul selama pelaksanaan proyek, berbagai strategi solutif diterapkan untuk memastikan kelancaran proyek dan tetap tercapainya tujuan utama. Pendekatan yang dilakukan bersifat fleksibel dan adaptif terhadap kondisi lapangan serta mempertimbangkan ketersediaan sumber daya yang ada. Berikut adalah uraian solusi yang diimplementasikan:

1. Penyesuaian Jadwal Pertemuan dan Optimalisasi Waktu Diskusi Langsung

Untuk mengatasi keterbatasan jadwal mitra GMLS, penyesuaian dilakukan secara fleksibel dengan mengatur pertemuan tatap muka di waktu yang tersedia bagi mitra, seperti di sore hari setelah kegiatan lapangan atau pada akhir pekan. Penjadwalan dilakukan berdasarkan koordinasi langsung dengan perwakilan GMLS agar waktu yang disepakati tidak mengganggu kegiatan utama mereka. Meskipun waktu diskusi cenderung singkat, setiap pertemuan dimaksimalkan untuk membahas poin-poin penting secara terfokus.

2. Memilih Lokasi Presentasi dengan Sinyal Stabil

Untuk memastikan presentasi chatbot dapat berjalan lancar, lokasi pertemuan dengan pihak GMLS dipilih berdasarkan pertimbangan kekuatan sinyal internet. Namun, sebagai antisipasi terhadap gangguan koneksi, juga menyiapkan dokumentasi alternatif seperti tangkapan layar chatbot. Dengan demikian, meskipun koneksi tidak optimal, pihak mitra tetap dapat memahami fungsi dan alur kerja

chatbot secara utuh. Pendekatan ini terbukti efektif dalam menjaga kelancaran penyampaian informasi teknis di lapangan.

3. Penyesuaian Tampilan Chatbot terhadap Desain Website Baru

Setelah terjadi pembaruan desain website GMLS, struktur tampilan chatbot diperbaharui agar tetap kompatibel dengan halaman baru. Elemen-elemen visual seperti warna, posisi ikon, dan gaya font disesuaikan dengan gaya baru website.