

BAB III

PELAKSANAAN KERJA

3.1 Kedudukan dan Koordinasi

3.1.1 Kedudukan

Selama menjalani program magang di Groovy Event Organizer, mahasiswa ditempatkan pada divisi *Research and Development* (R&D), yaitu divisi yang berperan sebagai pusat pengembangan teknologi internal perusahaan. Divisi ini bertanggung jawab dalam menyediakan berbagai solusi digital untuk mendukung operasional event, seperti sistem *chatbot*, antarmuka permainan interaktif, serta pengolahan visual berbasis *Artificial Intelligence* (AI). Dalam pelaksanaan tugasnya, mahasiswa berada di bawah koordinasi Head of Research and Development serta Wakil Kepala Divisi R&D yang berperan dalam mengarahkan jalannya proyek, mengatur alur kerja, dan memastikan setiap aktivitas berjalan sesuai standar teknis perusahaan.

Struktur kerja internal divisi R&D terdiri dari beberapa subkelompok yang dibentuk berdasarkan kebutuhan proyek dan keahlian teknis. Subkelompok tersebut meliputi tim pengembangan *chatbot* untuk keperluan event, tim *AI Photobooth* yang menangani sistem berbasis machine learning untuk pengolahan visual, serta tim pendukung lain yang berfokus pada modul digital tertentu. Pembagian kerja ini menciptakan lingkungan kerja yang kolaboratif dan dinamis, di mana setiap anggota tim saling berkoordinasi dalam pengembangan solusi teknologi.

Pada tahap awal pelaksanaan magang, mahasiswa ditempatkan pada subkelompok *chatbot* dengan tujuan memahami struktur sistem dan alur kerja yang digunakan perusahaan. Aktivitas yang dilakukan meliputi pemahaman logika percakapan, alur interaksi pengguna, serta integrasi sistem dengan

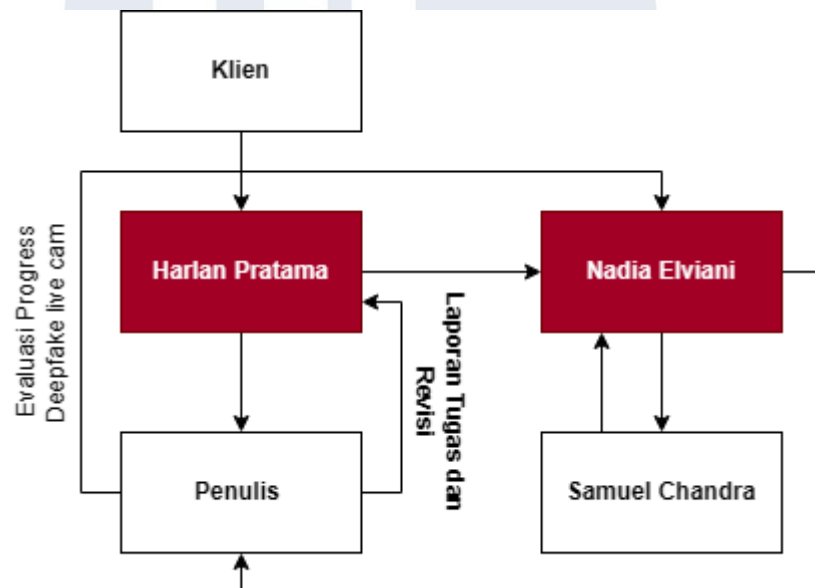
database yang tersedia. Berdasarkan evaluasi kompetensi oleh pembimbing lapangan, mahasiswa kemudian dialihkan ke subkelompok yang menangani pengembangan antarmuka pengguna (UI/UX), khususnya untuk proyek event interaktif seperti *Hunter Game*, *Scream Wall*, dan *Wishing Wall*. Pada tahap ini, mahasiswa berperan dalam perancangan tampilan antarmuka, pengaturan struktur halaman, serta memastikan kesesuaian tampilan dengan konsep dan kebutuhan *event*.

Selain pengembangan antarmuka, mahasiswa juga diberikan tugas untuk membantu proses instalasi dan konfigurasi sistem berbasis *Artificial Intelligence*, yaitu *deep live cam*, yang digunakan untuk kebutuhan internal perusahaan. Kegiatan ini meliputi instalasi aplikasi, pengaturan modul pendukung, penyesuaian parameter visual, serta pengujian kestabilan sistem dalam mode siaran langsung. Melalui kegiatan tersebut, mahasiswa memperoleh pemahaman mengenai penerapan teknologi visual generatif dalam konteks operasional.

Pada tahap selanjutnya, mahasiswa kembali dilibatkan dalam tim *chatbot* dengan peran sebagai *Quality Assurance (QA)*. Dalam posisi ini, mahasiswa bertanggung jawab melakukan pengujian fitur *chatbot*, memverifikasi alur percakapan, mengidentifikasi *bug*, serta mendokumentasikan hasil pengujian untuk dikoordinasikan dengan tim pengembang. Melalui kedudukan yang bersifat adaptif ini, mahasiswa memperoleh pemahaman menyeluruh mengenai siklus pengembangan sistem digital, mulai dari perancangan, implementasi, hingga pengujian, yang mencerminkan karakteristik kerja divisi R&D dalam industri *event organizer* berbasis teknologi.

3.1.2 Koordinasi

Alur koordinasi pekerjaan selama pelaksanaan magang di divisi *Research and Development (R&D)* berlangsung secara terstruktur dan bertingkat dengan pembimbing lapangan sebagai pusat koordinasi utama. Proses dimulai dari klien yang menyampaikan kebutuhan proyek, permintaan fitur, atau pembaruan sistem yang diperlukan untuk mendukung penyelenggaraan *event*. Seluruh kebutuhan tersebut terlebih dahulu diterima dan dianalisis oleh *Head of Research and Development* untuk diterjemahkan menjadi arahan teknis yang jelas sebelum didistribusikan kepada anggota tim.



Gambar 3.1 Bagan Alur Koordinasi

Setelah kebutuhan klien dipetakan menjadi penugasan teknis, pembimbing lapangan mengalokasikan tugas kepada mahasiswa sesuai dengan kompetensi yang dimiliki. Pada tahap ini, mahasiswa bertanggung jawab menyelesaikan pekerjaan seperti pengembangan antarmuka berbasis HTML, CSS, dan JavaScript, penyesuaian UI/UX, integrasi dasar dengan backend, pengujian *chatbot*, serta penyusunan dokumentasi pendukung. Selama proses pengerjaan, mahasiswa melakukan koordinasi secara berkala

dengan pembimbing untuk memastikan kesesuaian fungsi, alur interaksi, dan standar teknis yang berlaku di divisi R&D.

Setelah output awal diselesaikan, hasil pekerjaan diserahkan kembali kepada pembimbing lapangan untuk dilakukan pengecekan awal. Apabila pekerjaan telah memenuhi standar dasar, hasil tersebut diteruskan kepada wakil kepala divisi R&D untuk dilakukan evaluasi lanjutan. Tahap evaluasi ini bertujuan memastikan konsistensi kualitas, kerapian struktur, serta kesiapan teknis sebelum pekerjaan dinyatakan layak digunakan. Jika ditemukan kekurangan, instruksi revisi disampaikan kembali kepada mahasiswa untuk dilakukan perbaikan sesuai arahan yang diberikan.

Pada proyek tertentu yang membutuhkan integrasi dengan modul teknis lain, koordinasi juga dilakukan dengan anggota divisi R&D yang memiliki tanggung jawab pada pengembangan sistem terkait, seperti modul *chatbot* atau *backend*. Hasil integrasi tersebut kemudian dikembalikan kepada mahasiswa untuk dilakukan pengecekan ulang sebelum diserahkan kembali kepada pembimbing lapangan untuk evaluasi akhir.

Apabila seluruh output telah memenuhi standar kualitas yang ditetapkan divisi R&D, pembimbing lapangan meneruskan hasil akhir kepada pihak klien. Pola koordinasi bertingkat ini memastikan setiap pekerjaan melalui proses validasi yang sistematis dan menyeluruh, sekaligus mencerminkan pentingnya kolaborasi tim dan peran pembimbing sebagai penghubung utama antara kebutuhan klien dan tim internal perusahaan.

3.2 Tugas yang Dilakukan

Tabel 3.1 Detail Pekerjaan yang Dilakukan

No.	Minggu	Proyek	Keterangan
-----	--------	--------	------------

1	Minggu 1 -2 (8 - 22 Agustus)	Orientasi awal	Pengenalan <i>workflow</i> proyek, setup lingkungan kerja, serta mempelajari penggunaan Firebase dasar sebagai pendukung sistem yang digunakan perusahaan.
2	Minggu 2-5 (22 Agustus - 5 September)	<i>Wishing Wall</i> untuk <i>event Gathering</i> PT XYZ 1	Mengembangkan <i>backend</i> dan <i>frontend Wishing Wall</i> menggunakan Visual Studio Code dan Firebase untuk pengiriman serta penampilan pesan secara real-time. Sistem diuji dan dioptimalkan agar proses input dan display berjalan lancar serta data dapat diekspor dengan rapi.
3	Minggu 6-8	<i>Hunter Game</i> untuk PT. XYZ 2	Merancang elemen antarmuka dan struktur halaman untuk sistem <i>Hunter Game</i> , kemudian memastikan seluruh komponen visual responsif dan dapat digunakan pada perangkat yang berbeda. Pengujian fungsi dasar juga dilakukan untuk memastikan kelancaran penggunaan saat <i>event</i> .
4	Minggu 9- 13	<i>Scream Wall</i> untuk PT. XYZ 2	Membuat <i>layout</i> antarmuka <i>Scream Wall</i> serta melakukan penyesuaian visual berdasarkan instruksi pembimbing dan kebutuhan klien. Selain itu, dilakukan verifikasi fungsi agar sistem

			dapat menampilkan output visual secara real time.
5	Minggu 14-17	<i>Wishing Wall</i> untuk <i>event Awarding PT XYZ 3</i>	Melakukan penyesuaian desain <i>Wishing Wall</i> untuk kebutuhan <i>event</i> berbeda, termasuk pembaruan struktur halaman dan elemen visual. Pekerjaan juga meliputi revisi minor sesuai hasil evaluasi pembimbing.
6	Minggu 18-20	<i>Quality Assurance</i> firebase html dan <i>Chatbot n8n</i>	Bertanggung jawab melakukan pengujian fungsi pada sistem berbasis Firebase dan <i>chatbot</i> yang dibuat menggunakan n8n, mencatat bug, serta memastikan alur percakapan berjalan sesuai skenario. Setiap temuan dilaporkan kepada pembimbing untuk proses revisi lanjutan.
7	Minggu 19-20	Penginstallan dan implementasi <i>Deepfake live cam AI</i>	Melakukan instalasi perangkat lunak <i>Deepfake Live Cam</i> serta konfigurasi awal untuk keperluan internal event. Proses mencakup pengujian performa, identifikasi kendala teknis, dan pelaporan hasil evaluasi kepada pembimbing.

Tabel 3.1 menunjukkan rangkaian tugas yang dikerjakan selama pelaksanaan magang di PT. Sahabat Pesta Indonesia. Selama periode magang, dalam berbagai proyek pengembangan sistem interaktif yang digunakan

untuk mendukung penyelenggaraan event klien, dengan fokus utama pada pengembangan antarmuka, integrasi sistem, serta pengujian kualitas aplikasi.

Pada tahap awal magang, yaitu minggu pertama hingga minggu kedua, dilaksanakan orientasi awal yang mencakup pengenalan *workflow* divisi *Research & Development* (R&D), pemahaman alur kerja proyek, serta setup lingkungan kerja. Pada fase ini, dipelajari penggunaan Firebase dasar yang digunakan sebagai backend pendukung sistem internal perusahaan, termasuk struktur database dan mekanisme pengambilan data.

Selanjutnya, pada minggu kedua hingga minggu kelima, dilakukan pengembangan *Wishing Wall* untuk *event Gathering* PT XYZ 1. Dalam proyek ini, diberikan bertanggung jawab mengembangkan frontend dan backend sederhana menggunakan HTML, CSS, dan Firebase. Tugas yang dilakukan meliputi pembuatan tampilan antarmuka, pengaturan alur input pesan pengguna, integrasi data ke Firebase, serta pengujian sistem agar pesan dapat ditampilkan secara real-time dan diekspor dengan rapi setelah acara berlangsung.

Pada minggu keenam hingga minggu kedelapan, dikerjakan Proyek *Hunter Game* untuk PT XYZ 2. Terdapat hal yang difokuskan pada perancangan struktur halaman dan elemen antarmuka permainan, termasuk pengaturan layout dan responsivitas tampilan agar dapat digunakan pada berbagai perangkat. Selain itu, dilakukan pengujian fungsi dasar untuk memastikan interaksi pengguna berjalan lancar saat digunakan pada event.

Masih pada rentang waktu tersebut, dilakukan pengembangan proyek *Scream Wall* untuk PT XYZ 2 pada minggu kesembilan hingga minggu ketiga belas. Dalam proyek ini, diberikan tanggung jawab menyusun layout antarmuka, menyesuaikan elemen visual sesuai kebutuhan klien, serta melakukan verifikasi fungsi agar sistem mampu menampilkan output visual secara real-time berdasarkan input suara yang diterima.

Pada minggu keempat belas hingga minggu ketujuh belas, dikerjakan *Wishing Wall* untuk *event Awarding PT XYZ 3*. Tugas utama yang dilakukan meliputi penyesuaian desain antarmuka, pembaruan struktur halaman, serta revisi elemen visual berdasarkan evaluasi pembimbing. Tampilan sistem selaras dengan konsep acara serta tetap stabil saat digunakan. Tampilan sistem dipastikan selaras dengan konsep acara serta tetap stabil saat digunakan.

Memasuki minggu kedelapan belas hingga minggu kedua puluh, peran sebagai Quality Assurance dijalankan pada sistem berbasis Firebase dan chatbot yang dikembangkan menggunakan n8n. Pada tahap ini, pengujian alur percakapan chatbot dilakukan, fungsi sistem diverifikasi, bug yang ditemukan dicatat, serta hasil pengujian dilaporkan kepada pembimbing untuk proses perbaikan lanjutan.

Selain itu, pada minggu kesembilan belas hingga minggu kedua puluh, instalasi dan implementasi Deepfake Live Cam berbasis Artificial Intelligence juga dilakukan untuk kebutuhan internal perusahaan. [18], [19]. Tugas yang dilakukan mencakup instalasi perangkat lunak, konfigurasi awal sistem, pengujian performa, serta identifikasi kendala teknis yang kemudian dilaporkan kepada pembimbing lapangan sebagai bahan evaluasi.

Secara keseluruhan, rangkaian tugas tersebut memberikan pengalaman komprehensif dalam pengembangan sistem interaktif untuk event, mulai dari pembuatan antarmuka, integrasi backend, pengujian kualitas, hingga implementasi teknologi berbasis AI. Keterlibatan langsung dalam setiap proyek membantu pemahaman terhadap alur kerja divisi R&D serta meningkatkan kemampuan teknis dan koordinasi dalam lingkungan kerja profesional.

3.3 Uraian Pelaksanaan Kerja

Selama masa magang, keterlibatan dilakukan dalam enam proyek utama yang mencakup tahap perancangan, implementasi, penyesuaian visual, pengembangan logika fungsional, pengujian, hingga *deployment*. Setiap proyek memiliki karakteristik teknis yang berbeda, sehingga diperlukan penyesuaian metode kerja, pendekatan pemecahan masalah, serta koordinasi dengan supervisor agar hasil akhir sesuai dengan kebutuhan event dan standar perusahaan.

3.3.1 Proses Pelaksanaan

Pada proyek Wishing Wall untuk Event 1 dan Event 3, proses pengerjaan diawali dari tahap perancangan struktur tampilan menggunakan *HTML*, penyusunan gaya visual melalui *CSS*, serta implementasi interaksi dengan *JavaScript*. Pekerjaan kemudian dilanjutkan ke tahap integrasi *front end* dan *back end* agar sistem dapat menerima input pengguna dan menampilkannya dalam bentuk tampilan yang interaktif. Selain itu, proses *deployment* ke *hosting* juga dilakukan sehingga aplikasi dapat diakses secara *online*. Meskipun tidak ditemukan kendala besar, bagian yang memerlukan perhatian khusus adalah proses penyesuaian tampilan berdasarkan instruksi Head divisi RnD Harlan Pratama, yang sering membutuhkan *trial and error* untuk memastikan posisi kotak berisi nama dan ucapan tampil dengan proporsi serta peletakan yang tepat.

Pada proyek Hunter Game, mekanisme gameplay berupa pengumpulan lima gems dikembangkan melalui proses pemindaian gambar target. Tahap awal diawali dengan perancangan alur interaksi pemain serta memastikan sistem mampu mengenali dan memvalidasi gambar yang telah ditentukan. Selama proses pengerjaan, beberapa revisi diberikan oleh supervisor, terutama terkait penempatan kamera yang digunakan untuk scanning, pemilihan warna teks agar lebih mudah terlihat oleh pengguna, serta perbaikan penulisan pada gambar target untuk menghindari kebingungan. Seluruh revisi tersebut dilakukan melalui iterasi bertahap

hingga gameplay berjalan sesuai harapan dan memberikan pengalaman yang mulus bagi pengguna.

Pada proyek *Scream Wall*, fokus diarahkan pada pengembangan antarmuka yang responsif dan menarik, yang mencakup penyusunan *layout*, optimasi *background*, pemilihan gaya teks, serta pengaturan sensitivitas audio. Proses diawali dengan pembuatan *UI* dasar, kemudian dilanjutkan dengan pengaturan input suara agar sistem mampu menangkap teriakan pengguna sebagai pemicu. Meskipun tidak ditemukan kendala teknis yang signifikan, berbagai pengujian lapangan tetap dilakukan guna menyesuaikan sensitivitas suara agar sistem dapat bekerja secara optimal saat digunakan oleh banyak orang. Tujuan dari proses ini adalah memastikan pengguna benar-benar harus berteriak dengan intensitas tertentu untuk memicu aksi visual pada layar, sehingga pengalaman event tetap menarik dan interaktif.

Dalam tugas Chatbot *QA*, peran yang dijalankan berfokus pada pelaksanaan pengujian menyeluruh untuk memastikan stabilitas dan konsistensi respons chatbot. Proses pengujian dilakukan mulai dari input dasar hingga interaksi yang lebih kompleks untuk memantau respons sistem, terutama ketika menerima banyak perintah dalam waktu singkat. Error yang sering ditemukan meliputi respons yang tidak sesuai konteks, sistem yang berjalan lambat, serta kondisi ketika chatbot secara tiba-tiba mengirimkan *spam* respons. Untuk mengukur ketahanan sistem terhadap beban tinggi, digunakan metode *stress testing* [20], sehingga kemampuan chatbot dalam menghadapi jumlah permintaan yang besar atau permintaan berulang dalam waktu singkat dapat diamati. Hasil pengujian tersebut kemudian dijadikan dasar untuk perbaikan lanjutan guna meningkatkan stabilitas performa.

Pada proyek *Deepfake Live Cam*, pekerjaan diawali dengan penyiapan lingkungan pengembangan menggunakan *CMD*, *PowerShell*, *Python*, *Git*, dan *Conda*. Tahap instalasi melibatkan konfigurasi berbagai *library* pendukung sistem *deepfake*, namun ditemukan beberapa kendala

berupa ketidakcocokan versi *library* dengan *CUDA* serta keterbatasan dukungan terhadap *graphic card* yang digunakan karena termasuk generasi baru. Setelah dilakukan beberapa penyesuaian dan percobaan, aplikasi berhasil dijalankan, meskipun belum optimal karena sistem hanya dapat berjalan menggunakan *CPU*. Kondisi ini menyebabkan nilai *FPS* menjadi rendah sehingga pengalaman penggunaan tidak semulus yang diharapkan. Meskipun demikian, hasil akhir yang diserahkan tetap memenuhi kebutuhan dasar untuk keperluan demonstrasi internal.

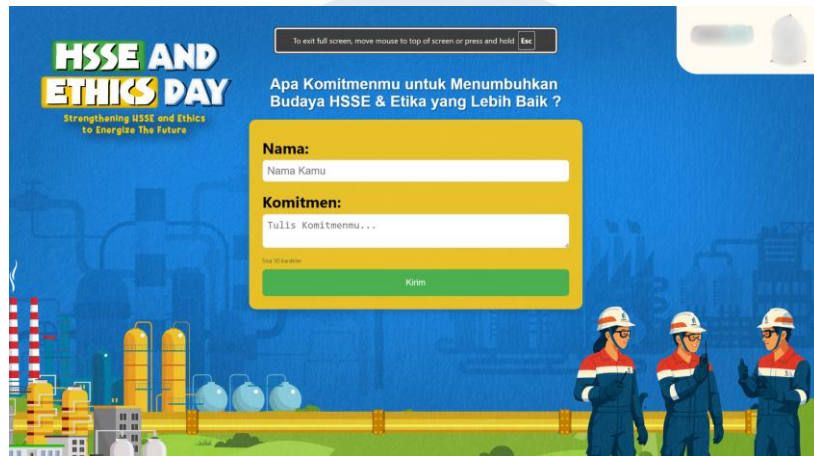
Melalui seluruh rangkaian proyek tersebut, diperoleh pengalaman menyeluruh mengenai proses pengembangan aplikasi interaktif, mulai dari tahap perancangan, penyusunan kode, penanganan revisi, identifikasi kendala teknis, hingga proses pengujian dan *deployment*. Setiap proyek memberikan tantangan yang berbeda dan berkontribusi dalam memperkuat kemampuan *problem solving*, kolaborasi, serta implementasi teknis yang sesuai dengan standar industri.

3.3.1.1 Wishing Wall untuk event Gathering PT XYZ 1

Proyek pertama yang dikerjakan yaitu Wishing Wall yang merupakan salah satu proyek yang dikerjakan oleh mahasiswa selama program magang PT. Sahabat Pesta Indonesia untuk event PT XYZ 1. Proyek ini bertujuan untuk mendukung aktivitas *engagement* pada event Gathering PT XYZ 1 melalui penyediaan sistem interaktif berbasis web yang memungkinkan peserta acara mengirimkan nama dan ucapan secara real time. Sistem ini dirancang agar pesan yang dikirim oleh peserta dapat ditampilkan secara langsung pada layar LED selama acara berlangsung, sehingga menciptakan interaksi yang lebih aktif antara peserta dan penyelenggara event.

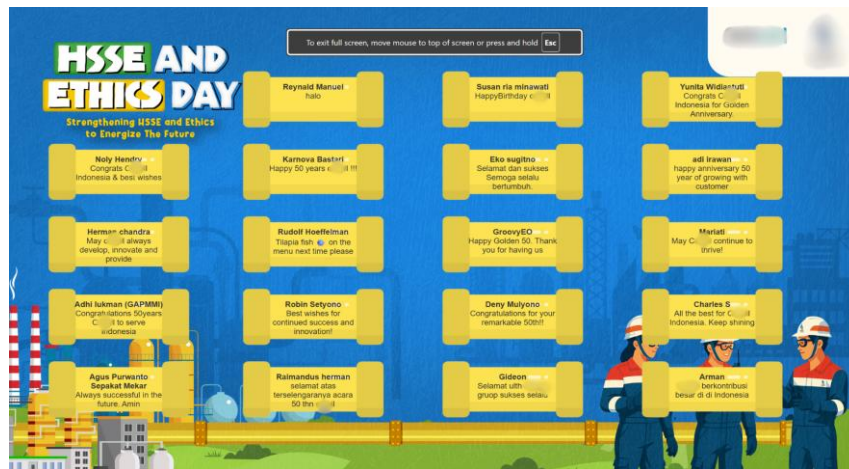
Tahap awal pengerjaan proyek dimulai dengan diskusi kebutuhan bersama supervisor untuk menentukan konsep sistem, alur interaksi pengguna, serta kebutuhan teknis yang harus dipenuhi. Pada tahap ini, mahasiswa bertugas memahami tujuan penggunaan Wishing Wall dalam

konteks event, termasuk batasan tampilan pada layar LED, kebutuhan kestabilan sistem, serta kemudahan penggunaan bagi peserta acara. Berdasarkan hasil diskusi tersebut, mahasiswa menyusun rancangan alur sistem yang mencakup proses input data, penyimpanan data, dan penampilan pesan secara real time.



Gambar 3.2 Halaman Input Pengiriman Ucapan pada Sistem Wishing Wall PT XYZ 1

Gambar 3.2 menunjukkan tampilan halaman input yang dirancang dan dikembangkan oleh mahasiswa sebagai media bagi peserta untuk mengirimkan ucapan. Pada tahap ini, mahasiswa bertanggung jawab mengatur struktur layout halaman menggunakan pendekatan berbasis container agar elemen seperti nama pengirim, isi pesan, dan tombol kontrol tersusun secara seimbang. Mahasiswa juga melakukan beberapa kali penyesuaian ukuran elemen dan posisi box ucapan melalui proses trial and error untuk memastikan tampilan tetap rapi dan proporsional ketika digunakan pada layar LED berukuran besar.



Gambar 3.3 Halaman Output Penayangan Ucapan pada Sistem Wishing Wall PT XYZ 1

Dari gambar 3.3 pada halaman output, diimplementasikan tampilan khusus yang digunakan untuk menampilkan pesan pada layar LED. Dilakukan pengujian pada berbagai resolusi layar untuk memastikan layout tetap konsisten saat ditampilkan pada LED, monitor operator, maupun perangkat lain.

Transisi kemunculan pesan baru juga diuji agar animasi berjalan halus tanpa menyebabkan lag atau keterlambatan tampilan, sehingga tidak mengganggu jalannya acara.

Setelah struktur tampilan ditentukan, mahasiswa melanjutkan dengan implementasi fungsi pengelolaan data. Mahasiswa menyusun struktur HTML sebagai kerangka halaman input yang mencakup judul halaman, form input nama, textarea untuk pesan, indikator jumlah karakter, serta tombol “Kirim” dan “Hapus Ucapan”.

```

<title>Kirim Ucapan</title>
<link rel="stylesheet" href="style.css">
</head>
<body>

<button onclick="hapusSemuaUcapan()" id="hapusBtn">Hapus Ucapan</button>

<h1>Apa Komitmenmu untuk Menumbuhkan <br>
Budaya HSSE & Etika yang Lebih Baik ?</h1>
<form id="ucapanForm">
  <label for="nama">Nama:</label>
  <input type="text" id="nama" placeholder="Nama kamu" required>

  <label for="pesan">Komitmen:</label>
  <textarea id="pesan" placeholder="Tulis Komitmenmu..." required></textarea>
  <small id="counter">Sisa 50 karakter</small><br>

  <button type="submit">Kirim</button>
</form>

<script src="https://www.gstatic.com/firebasejs/9.22.0/firebase-app-compat.js"></script>
<script src="https://www.gstatic.com/firebasejs/9.22.0/firebase-database-compat.js"></script>

```

Gambar 3.4 Struktur Halaman Input Sistem Wishing Wall PT XYZ 1

Gambar 3.4 menampilkan potongan kode HTML yang digunakan sebagai dasar struktur halaman input. Pada tahap ini, mahasiswa mengintegrasikan halaman dengan Firebase Realtime Database dengan memuat library Firebase agar data ucapan yang dikirim dapat disimpan dan dipanggil kembali secara real time. Seluruh logika pengiriman data, validasi input, serta fungsi penghapusan data dijalankan melalui file JavaScript yang terpisah, sehingga struktur kode tetap terorganisasi dan mudah dikelola.

Untuk mendukung kenyamanan visual dan konsistensi tampilan, mahasiswa melakukan pengaturan antarmuka menggunakan CSS. Pengaturan ini bertujuan agar aplikasi tetap stabil dan mudah dibaca pada berbagai ukuran layar.

```

body {
  font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen,
  background: url('vbg.png') no-repeat center center fixed;
  background-size: cover;

  /* TAMBAHKAN/UBAH BAGIAN INI */
  margin: 0;
  min-height: 80vh; /* Tinggi minimal sama dengan tinggi layar */
  display: flex; /* Mengaktifkan mode Flexbox */
  flex-direction: column; /* Mengatur item berurutan ke bawah */
  justify-content: center; /* Menengahkan secara vertikal */
  align-items: center; /* Menengahkan secara horizontal */
  padding: 20px; /* Jarak aman di tepi layar */
  box-sizing: border-box;
  content: flex-start;
}

```


Gambar 3.5 Pengaturan Antarmuka Sistem Wishing Wall PT XYZ 1

Pada tahap gambar 3.5, mahasiswa mengatur elemen body dengan menerapkan background image serta penggunaan flexbox untuk memusatkan konten secara horizontal dan vertikal. Properti seperti padding, min-height, dan box-sizing digunakan untuk menjaga kestabilan layout, sehingga tampilan tidak berubah secara signifikan ketika aplikasi dibuka pada perangkat dengan resolusi yang berbeda.

```
h1 {  
  margin-bottom: 40px;  
  color: white;  
  text-shadow: 1px 1px 3px rgba(0,0,0,0.5); /* Bayangan teks lebih halus */  
  font-family: 'Montserrat', sans-serif;  
  font-size: 2.8em; /* Ukuran font untuk desktop */  
  align-items: center;  
}  
  
form {  
  display: inline-block;  
  text-align: left;  
  background: rgba(255, 201, 22, 0.9); /* Latar belakang putih transparan */  
  
  /* Padding yang lebih wajar untuk desktop */  
  padding: 28px;  
  
  border-radius: 15px;  
  box-shadow: 0 4px 15px rgba(0,0,0,0.2);  
  
  /* Lebar maksimal form di desktop */  
  width: 100%;  
  max-width: 700px;  
  box-sizing: border-box; /* Padding tidak akan menambah lebar total */  
}
```

Gambar 3.6 Pengaturan Header dan Form Input pada Sistem Wishing Wall PT XYZ 1

Pada Gambar 3.6 mahasiswa mengatur elemen judul dengan ukuran font yang besar, warna kontras, serta efek bayangan ringan agar tetap terbaca jelas pada latar belakang yang digunakan dalam event.


```

form {
  display: inline-block;
  text-align: left;
  background: rgba(255, 201, 22, 0.9); /* Latar belakang putih transparan */

  /* Padding yang lebih wajar untuk desktop */
  padding: 28px;

  border-radius: 15px;
  box-shadow: 0 4px 15px rgba(0,0,0,0.2);

  /* Lebar maksimal form di desktop */
  width: 100%;
  max-width: 700px;
  box-sizing: border-box; /* Padding tidak akan menambah lebar total */
}

label {
  font-size: 2em; /* TAMBAHKAN properti ini */
  display: block;
  margin-top: 10px;
  font-weight: bold;
}

input, textarea {
  width: 100%;
  padding: 10px;
  margin-top: 5px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  border-radius: 8px;
  font-size: 22px;
  box-sizing: border-box; /* Padding tidak akan menambah lebar total */
}

```

Gambar 3.7 Kode untuk Tampilan *Label* dan *Input* Text PT XYZ 1

Pada bagian form, mahasiswa menerapkan latar belakang semi-transparan, sudut melengkung, serta bayangan ringan untuk menonjolkan area input. Elemen input dan textarea dirancang dengan lebar penuh dan ukuran font yang cukup besar agar mudah digunakan oleh peserta, termasuk pada perangkat mobile.

Selain pengaturan tampilan utama, mahasiswa juga mengatur fungsi kontrol dan responsivitas tampilan agar sistem dapat digunakan secara optimal selama acara berlangsung.

```

button[type="submit"] {
  width: 100%; /* Tombol kirim memenuhi lebar form */
  padding: 18px 25px;
  background-color: #4CAF50;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  font-size: 18px;
  margin-top: 10px;
}

button[type="submit"]:hover {
  background-color: #45a049;
}

/* Tombol hapus semua ucapan (disembunyikan) */
#hapusBtn {
  position: fixed;
  bottom: 20px;
  right: 20px;
  z-index: 999;
  padding: 8px 12px;
  background-color: #f44336;
  color: white;
  border: none;
  border-radius: 50%;
  cursor: pointer;
  box-shadow: 0 2px 6px rgba(0,0,0,0.3);
  display: none;
}

```

Gambar 3.8 Pengaturan Tombol Kirim dan Kontrol Admin pada Sistem Wishing Wall PT XYZ 1

Tombol “Kirim” dirancang memenuhi lebar form untuk memudahkan interaksi pengguna, terutama pada perangkat dengan layar kecil. Tombol “Hapus Ucapan” ditempatkan secara fixed di sudut kanan bawah dan disembunyikan secara default, karena hanya digunakan oleh admin melalui kombinasi tombol tertentu.

```

/* ===== INI BAGIAN PENTING UNTUK TAMPILAN HP ===== */
@media (max-width: 600px) {
  body {
    padding: 20px; /* Jarak di sekeliling body diperkecil */
  }

  h1 {
    font-size: 1.6em; /* Ukuran judul diperkecil */
  }

  form {
    padding: 20px; /* Jarak di dalam form diperkecil */
  }
}

/* ===== INI BAGIAN BARU UNTUK SEMUA TAMPILAN LANDSCAPE (HP & iPad) ===== */
@media (orientation: landscape) {
  body {
    /* Pindahkan konten ke atas, jangan paksa di tengah */
    justify-content: center;
    padding-top: 50px; /* Beri sedikit jarak dari atas */
    padding-bottom: 50px;
  }

  h1 {
    font-size: 2em; /* Ukuran judul yang pas untuk landscape */
    margin-bottom: 20px;
  }
}

```

Gambar 3.9 Pengaturan Responsivitas Sistem Wishing Wall pada Perangkat Mobile

Pada gambar 3.9 mahasiswa menambahkan media query untuk menyesuaikan tampilan pada layar ponsel dan tablet, sehingga elemen visual tetap proporsional dan mudah dibaca tanpa mengurangi kenyamanan pengguna.

Tahap selanjutnya adalah pengujian sistem. Mahasiswa melakukan pengujian dengan mengirimkan banyak pesan dalam waktu yang relatif singkat untuk memastikan sistem mampu menampung input secara bersamaan tanpa mengalami error, lag, atau keterlambatan tampilan. Pengujian juga dilakukan pada berbagai resolusi layar untuk memastikan konsistensi tampilan pada layar LED, monitor operator, dan perangkat lainnya.

Setelah seluruh fungsi berjalan stabil, mahasiswa melakukan deployment aplikasi ke server online agar sistem dapat diakses oleh tim event tanpa bergantung pada server lokal. Tahap ini dilakukan sebagai persiapan akhir sebelum acara berlangsung untuk memastikan sistem siap digunakan pada hari pelaksanaan event.

Melalui proyek Wishing Wall ini, mahasiswa memperoleh pengalaman langsung dalam mengembangkan aplikasi interaktif untuk kebutuhan event, mulai dari perancangan konsep, implementasi sistem, pengujian fungsional, hingga persiapan teknis sebelum digunakan secara langsung. Proyek ini memberikan pemahaman mengenai pentingnya kestabilan sistem, konsistensi tampilan, serta kesiapan teknis dalam mendukung pelaksanaan event berskala besar.

3.3.1.2 Hunter Game untuk PT. XYZ 2

Proyek Hunter Game merupakan proyek kedua yang dikerjakan selama program magang di PT XYZ 2. Proyek ini

dikembangkan sebagai aplikasi permainan interaktif yang digunakan untuk mendukung aktivitas engagement pada salah satu event perusahaan. Melalui permainan ini, peserta diajak untuk berpartisipasi secara aktif dengan cara mengumpulkan lima item atau *gems* melalui proses pemindaian gambar target yang telah ditentukan oleh panitia event.

Pada permainan ini, diberikan tanggung jawab dalam merancang alur permainan, mengembangkan antarmuka aplikasi, serta mengimplementasikan logika permainan agar dapat berjalan secara stabil dan responsif selama event berlangsung. Aplikasi dirancang agar mudah digunakan oleh peserta dari berbagai kalangan dan dapat diakses menggunakan berbagai perangkat, terutama smartphone yang menjadi perangkat utama selama acara.

Tahap awal pengerjaan proyek dimulai dengan analisis kebutuhan sistem dan alur permainan. Pada tahap ini, disusun konsep permainan mulai dari tahap awal permainan hingga kondisi akhir ketika seluruh *gems* berhasil dikumpulkan. Selain itu, ditentukan juga elemen visual yang perlu ditampilkan pada setiap tahap permainan, termasuk halaman awal, halaman pemindaian, notifikasi penemuan *gem*, dan halaman hasil akhir.

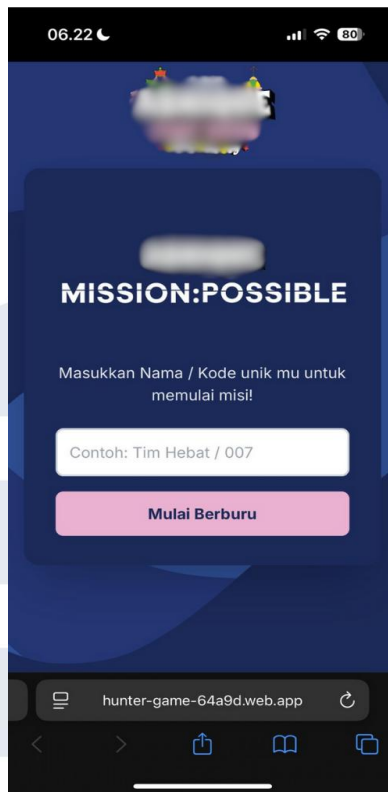
Setelah konsep dan alur permainan ditetapkan, dilanjutkan dengan perancangan struktur halaman dan antarmuka pengguna menggunakan HTML dan CSS. Antarmuka dirancang agar sederhana dan intuitif, sehingga peserta dapat langsung memahami cara bermain tanpa memerlukan penjelasan tambahan. Pada tahap ini juga dipastikan setiap elemen UI bersifat responsif agar tampilan dapat menyesuaikan dengan ukuran layar laptop maupun smartphone.

Tahap berikutnya adalah implementasi logika permainan menggunakan JavaScript. Pada tahap ini, diintegrasikan modul

pemindaian gambar yang disediakan oleh tim R&D untuk mendukung proses scanning menggunakan kamera perangkat. Selain itu, logika validasi gambar, perhitungan jumlah *gems* yang berhasil dikumpulkan, serta kondisi penyelesaian permainan (*game completion*) juga diimplementasikan. Sistem notifikasi turut ditambahkan sebagai umpan balik bagi pengguna ketika proses pemindaian berhasil maupun gagal.

Setelah proses implementasi selesai, pengujian sistem dilakukan secara berkala untuk memastikan permainan dapat berjalan dengan lancar tanpa kendala teknis. Pengujian dilakukan dengan mensimulasikan proses permainan secara berulang, termasuk pemindaian gambar, penambahan jumlah *gems*, serta perpindahan antar halaman, guna memastikan seluruh alur permainan berjalan sesuai dengan rancangan.

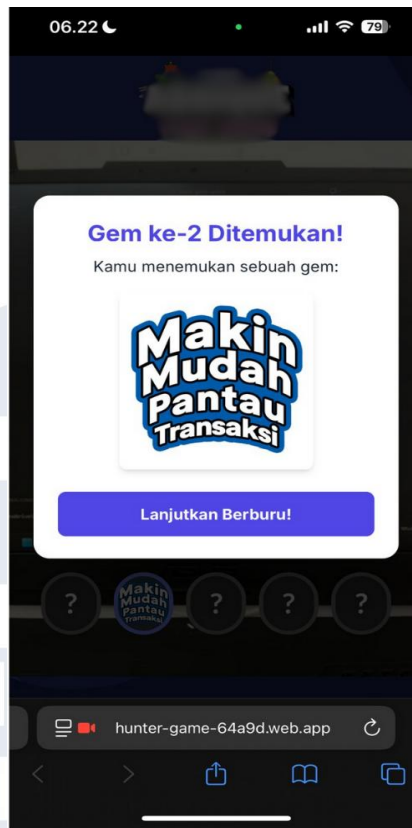
Aplikasi Hunter Game dibagi ke dalam beberapa halaman utama yang merepresentasikan tahapan permainan, mulai dari tahap awal hingga tahap akhir permainan. Pembagian halaman ini bertujuan untuk menampilkan alur permainan secara jelas dan terstruktur bagi peserta event.



Gambar 3.10 Halaman Awal Permainan Hunter Game pada Event PT XYZ 2

Halaman Awal Permainan Hunter Game pada Event PT XYZ 2 Gambar 3.10 menunjukkan halaman awal permainan yang digunakan sebagai titik masuk sebelum peserta memulai permainan. Pada halaman ini, disediakan form input sederhana untuk memasukkan nama atau kode unik peserta, serta tombol untuk memulai permainan. Desain halaman dibuat sederhana agar peserta dapat langsung memahami konteks permainan dan melanjutkan ke tahap berikutnya tanpa kebingungan.

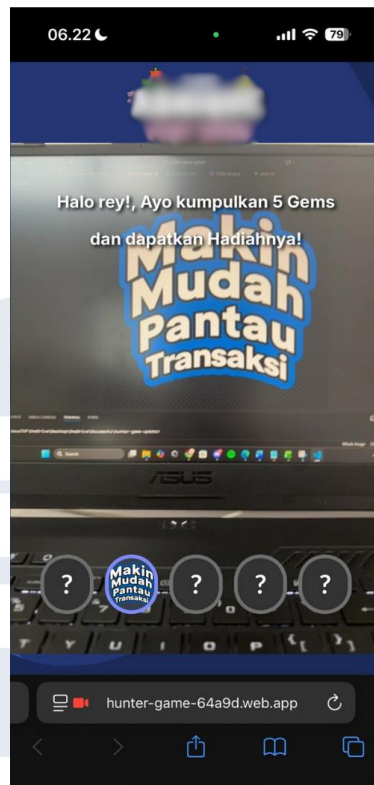
Setelah peserta memulai permainan, sistem akan menampilkan halaman pemindaian yang mengaktifkan kamera perangkat untuk mencari dan memindai gambar target.



Gambar 3.11 Halaman Pemindaian Gambar Target pada Hunter Game PT XYZ 2

Gambar 3.11 memperlihatkan halaman pemindaian yang digunakan peserta untuk menemukan gems. Pada halaman ini, ditampilkan instruksi singkat serta indikator jumlah gems yang telah dikumpulkan. Tampilan kamera ditampilkan secara real time untuk mendukung proses scanning selama permainan berlangsung.

UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.12 Notifikasi Penemuan Gem pada Hunter Game PT XYZ 2

Gambar 3.12 menunjukkan tampilan notifikasi yang muncul ketika pengguna berhasil menemukan salah satu gem. Notifikasi ini menampilkan informasi jumlah gem yang telah dikumpulkan serta tombol untuk melanjutkan permainan, sehingga peserta tetap termotivasi untuk menyelesaikan misi.

Apabila seluruh lima gems telah berhasil dikumpulkan, sistem akan menampilkan halaman hasil akhir permainan.



Gambar 3.13 Halaman Hasil Akhir Hunter Game setelah Seluruh Gem Ditemukan

Gambar 3.13 menampilkan halaman akhir permainan yang menandakan bahwa misi telah selesai. Pada halaman ini ditampilkan pesan keberhasilan serta visualisasi seluruh gems yang telah dikumpulkan. Halaman ini juga berfungsi sebagai penutup alur permainan dan memastikan bahwa data keberhasilan peserta telah tercatat di dalam sistem.

Pada tahap pengembangan antarmuka permainan Hunter Game, diberikan tanggung jawab dalam mengimplementasikan tampilan dan interaksi permainan berbasis *augmented reality* (AR) menggunakan kombinasi teknologi A-Frame, MindAR, Tailwind CSS, serta struktur HTML. Pemilihan teknologi ini dilakukan untuk mendukung kebutuhan permainan yang memanfaatkan kamera perangkat sebagai media utama dalam proses pemindaian gambar target selama event berlangsung.

Dalam pengembangan sistem, digunakan file HTML sebagai kerangka utama aplikasi yang mengatur alur halaman, mulai dari halaman awal permainan, area pemindaian, tampilan notifikasi, hingga halaman hasil akhir permainan. Selain berfungsi sebagai struktur tampilan, file HTML juga digunakan sebagai titik awal inisialisasi modul AR yang memungkinkan sistem mendeteksi gambar target secara real time.



Gambar 3.14 Inisialisasi Library *Augmented Reality* pada Hunter Game

Gambar 3.14 menunjukkan bagian awal struktur aplikasi yang digunakan untuk menginisialisasi modul augmented reality. Pada tahap ini, diintegrasikan library A-Frame dan MindAR untuk mengaktifkan fitur image tracking melalui kamera perangkat. Inisialisasi ini menjadi dasar dari mekanisme utama permainan, karena memungkinkan sistem mengenali beberapa gambar target yang telah ditentukan sebagai *gems*.

Setelah inisialisasi AR dilakukan, diimplementasikan halaman awal permainan yang berfungsi untuk mengaitkan permainan dengan identitas peserta.

```

1 <!-- === Layar Login === -->
2 <div id="login-screen" class="fixed inset-0 flex items-center justify-center z-50 p-4 bg-cover bg-center"
3   style="background-image: url('your-background.jpg');">
4   <div class="rounded-2xl p-8 text-center shadow-2xl w-full max-w-sm"
5     style="background-color: #1b2a59; color: white;">
6
7     
8     <p class="mb-6 text-gray-200">Masukkan Nama / Kode unik mu untuk memulai misi!</p>
9
10    <input id="participant-id-input"
11      type="text"
12      placeholder="Contoh: Tim Hebat / 007"
13      class="w-full border-2 border-gray-300 rounded-lg p-3 mb-4 focus:border-pink-400 focus:ring-1 focus:ring-pink-400 outline-none text-gray-800">
14
15    <button id="start-button"
16      class="w-full font-bold py-3 px-4 rounded-lg focus:outline-none focus:ring-2 focus:ring-opacity-50 transition-transform transform hover:scale-105"
17      style="background-color: #eab2d0; color: #1b2a59;">
18      Mulai Berburu
19    </button>
20  </div>

```

Gambar 3.15 Struktur Halaman Awal Permainan Hunter Game

Gambar 3.15 memperlihatkan struktur halaman awal yang dirancang untuk menerima input nama atau kode unik peserta sebelum permainan dimulai. Halaman ini dilengkapi dengan tombol kontrol untuk memulai permainan, yang selanjutnya memicu perpindahan alur ke halaman pemindaian serta menjalankan logika permainan melalui JavaScript.

Tahap berikutnya adalah pengembangan halaman utama permainan, yang menampilkan area pemindaian serta indikator progres permainan.

```

1 <!-- === Kontainer Game === -->
2 <div id="game-container" class="hidden">
3   <div class="ui-container">
4     <!-- Bagian Atas (Header) -->
5     <div class="ui-header">
6       <h1 class="text-4xl font-bold mb-2 text-white" style="text-shadow: 2px 2px 4px #000;">GEMS HUNTER</h1>
7       <p id="scan-status" class="text-gray-200" style="text-shadow: 1px 1px 2px #000;">Arahkan kamera ke salah satu gem(s)</p>
8     </div>
9
10    <!-- Bagian Bawah (Footer dengan stempel) -->
11    <div class="ui-footer">
12      <div id="stamps-container" class="flex justify-center items-center gap-3">
13        <!-- Stempel akan di-generate oleh JavaScript -->
14      </div>
15    </div>
16  </div>

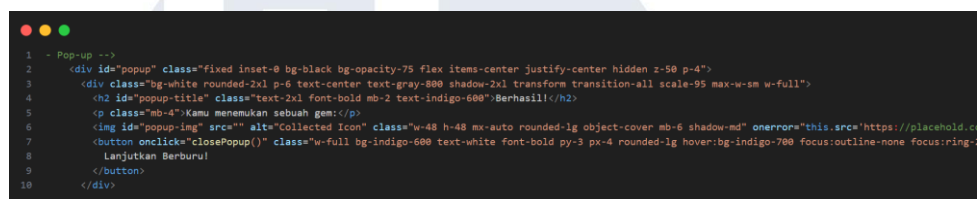
```

Gambar 3.16 Implementasi Struktur Antarmuka Kontainer Permainan

Gambar 3.16 menunjukkan struktur halaman permainan yang digunakan selama proses pemindaian gambar target. Pada halaman ini, ditampilkan instruksi singkat, area kamera untuk scanning, serta indikator jumlah *gems* yang telah dikumpulkan. Indikator progres dirancang agar dapat diperbarui secara dinamis melalui logika

permainan, sehingga peserta dapat memantau perkembangan permainan secara langsung.

Selain halaman utama permainan, dikembangkan juga komponen umpan balik visual berupa notifikasi dan halaman akhir permainan.

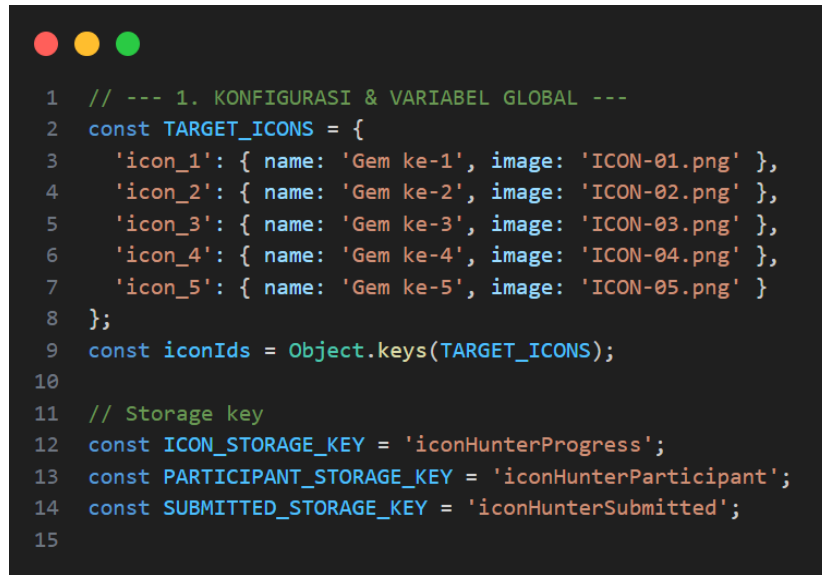


Gambar 3.17 Implementasi Komponen Popup Notifikasi dan Halaman Kemenangan

Gambar 3.17 menunjukkan struktur komponen notifikasi yang digunakan untuk memberikan umpan balik kepada pengguna ketika berhasil memindai gambar target. Komponen ini menampilkan informasi terkait *gem* yang ditemukan serta tombol untuk melanjutkan permainan. Apabila seluruh *gems* telah terkumpul, sistem akan menampilkan halaman hasil akhir sebagai penutup permainan dan menandai bahwa data keberhasilan peserta telah tersimpan di dalam sistem.

Untuk menjaga keteraturan sistem dan memudahkan pengelolaan aplikasi, diterapkan pemisahan antara struktur tampilan dan logika sistem. Logika permainan, pemrosesan AR, serta integrasi database ditempatkan pada berkas JavaScript terpisah, sementara HTML difokuskan sebagai kerangka antarmuka. Pendekatan ini membantu menjaga struktur aplikasi tetap rapi, terorganisasi, dan mudah dikembangkan lebih lanjut.

1. Konfigurasi Target Icons dan Variabel Global



```

1  // --- 1. KONFIGURASI & VARIABEL GLOBAL ---
2  const TARGET_ICONS = {
3    'icon_1': { name: 'Gem ke-1', image: 'ICON-01.png' },
4    'icon_2': { name: 'Gem ke-2', image: 'ICON-02.png' },
5    'icon_3': { name: 'Gem ke-3', image: 'ICON-03.png' },
6    'icon_4': { name: 'Gem ke-4', image: 'ICON-04.png' },
7    'icon_5': { name: 'Gem ke-5', image: 'ICON-05.png' }
8  };
9  const iconIds = Object.keys(TARGET_ICONS);
10
11 // Storage key
12 const ICON_STORAGE_KEY = 'iconHunterProgress';
13 const PARTICIPANT_STORAGE_KEY = 'iconHunterParticipant';
14 const SUBMITTED_STORAGE_KEY = 'iconHunterSubmitted';
15

```

Gambar 3.18 Implementasi Konfigurasi Target dan Variabel Global Permainan

Gambar 3.18 menunjukkan konfigurasi target dan variabel global yang disusun sebagai dasar logika permainan. Pada tahap ini, didefinisikan daftar gem yang harus ditemukan, serta mekanisme penyimpanan progress permainan untuk memastikan alur permainan berjalan sesuai perancangan.

2. Inisialisasi Firebase dan Pengaturan Koneksi Backend



```

1  // --- 2. INISIALISASI FIREBASE ---
2  import { initializeApp } from "https://www.gstatic.com/firebasejs/10.12.2/firebase-app.js";
3  import { getFirestore, doc, setDoc, serverTimestamp } from "https://www.gstatic.com/firebasejs/10.12.2/firebase-firestore.js";
4
5  const firebaseConfig = {
6    apiKey: "AIzaSyBghUe2JnEgJ9ZWEHEmnJMirJ0hX4nEGW4",
7    authDomain: "ucapan-balon.firebaseio.com",
8    databaseURL: "https://ucapan-balon-default-rtdb.asia-southeast1.firebaseio.com",
9    projectId: "ucapan-balon",
10   storageBucket: "ucapan-balon.appspot.com",
11   messagingSenderId: "72153682380",
12   appId: "1:72153682380:web:0e87ad71e05a8330108d69"
13 };
14
15 const app = initializeApp(firebaseConfig);
16 const db = getFirestore(app);

```

Gambar 3.19 Implementasi Inisialisasi Firebase dan Koneksi Database

Gambar 3.19 menunjukkan proses inisialisasi Firebase yang dilakukan untuk menghubungkan aplikasi Hunter Game dengan database backend. Pada tahap ini, dikonfigurasi juga koneksi database agar data penyelesaian permainan dapat disimpan secara terstruktur.

3. Fungsi Penyimpanan Data Penyelesaian Permainan

```
1 // --- 3. FIREBASE UTILS ---
2 async function saveCompletionToFirebase(participantId) {
3   const firebaseStatus = document.getElementById('firebase-status');
4   if (localStorage.getItem(SUBMITTED_STORAGE_KEY) === 'true') {
5     console.log('✅ Data sudah pernah dikirim.');
6     firebaseStatus.textContent = "Data sudah tersimpan sebelumnya! 🙌";
7     firebaseStatus.className = 'text-center text-sm mt-2 text-green-300';
8     return;
9   }
10
11   firebaseStatus.textContent = "Menyimpan data...";
12   firebaseStatus.className = 'text-center text-sm mt-2 text-yellow-300';
13
14   try {
15     const completionRef = doc(db, "messages", participantId);
16     await setDoc(completionRef, {
17       participantId,
18       completionTime: serverTimestamp(),
19       type: 'IconHunterCompletion'
20     });
21
22     localStorage.setItem(SUBMITTED_STORAGE_KEY, 'true');
23     firebaseStatus.textContent = "Data berhasil disimpan! 🎉";
24     firebaseStatus.classList.remove('text-yellow-300');
25     firebaseStatus.classList.add('text-green-300');
26     console.log('🔥 Data tersimpan ke Firebase.');
27   } catch (e) {
28     console.error("❌ Error Firebase:", e);
29     firebaseStatus.textContent = "Gagal menyimpan data. Cek koneksi internet.";
30     firebaseStatus.classList.remove('text-yellow-300');
31     firebaseStatus.classList.add('text-red-400');
32   }
33 }
34
```

Gambar 3.20 Fungsi Penyimpanan Data Penyelesaian Permainan ke Firebase

Gambar 3.20 menunjukkan fungsi `saveCompletionToFirebase` yang dikembangkan untuk menyimpan data penyelesaian permainan ke dalam database Firebase. Pada bagian ini, diatur mekanisme pengecekan data pada `localStorage` untuk mencegah pengiriman data ganda, serta mencatat waktu penyelesaian permainan menggunakan `server timestamp`. Implementasi fungsi ini memastikan bahwa setiap peserta yang berhasil menyelesaikan permainan Hunter Game tercatat secara otomatis dan terintegrasi dengan sistem backend.

4. Event Listener Utama untuk Mengelola Alur Login


```

1 // --- 4. EVENT LISTENER UTAMA ---
2 document.addEventListener('DOMContentLoaded', () => {
3   const loginScreen = document.getElementById('login-screen');
4   const gameContainer = document.getElementById('game-container');
5   const startButton = document.getElementById('start-button');
6   const participantInput = document.getElementById('participant-id-input');
7   const scene = document.querySelector('a-scene');
8
9   const storedParticipantId = localStorage.getItem(PARTICIPANT_STORAGE_KEY);
10
11   if (storedParticipantId) {
12     loginScreen.classList.add('hidden');
13     gameContainer.classList.remove('hidden');
14     initializeGame(storedParticipantId);
15     autoStartCamera(storedParticipantId); // auto start saat reload
16   } else {
17     loginScreen.classList.remove('hidden');
18     gameContainer.classList.add('hidden');
19   }
20 }

```

Gambar 3.21 Pengaturan Alur Login dan Inisialisasi Awal Permainan Hunter Game PT XYZ 2

Gambar 3.21 menunjukkan logika alur login dan permainan yang dirancang untuk mengatur mekanisme pengecekan identitas peserta agar sistem dapat menentukan apakah pengguna perlu melalui halaman login atau dapat langsung memulai permainan.

5. Fitur Reset Progress untuk Tujuan Debugging dan Maintenance

```

1 // Reset lewat klik logo
2 const titleElement = document.querySelector('.ui-header h1');
3 if (titleElement) {
4   let titleClickCount = 0;
5   let titleClickTimeout;
6   titleElement.addEventListener('click', () => {
7     titleClickCount++;
8     clearTimeout(titleClickTimeout);
9     titleClickTimeout = setTimeout(() => { titleClickCount = 0; }, 2000);
10    if (titleClickCount >= 5) {
11      localStorage.removeItem(PARTICIPANT_STORAGE_KEY);
12      localStorage.removeItem(SUBMITTED_STORAGE_KEY);
13      localStorage.removeItem(ICON_STORAGE_KEY);
14      alert("User ID & progress direset!");
15      setTimeout(() => window.location.reload(), 500);
16    }
17  });

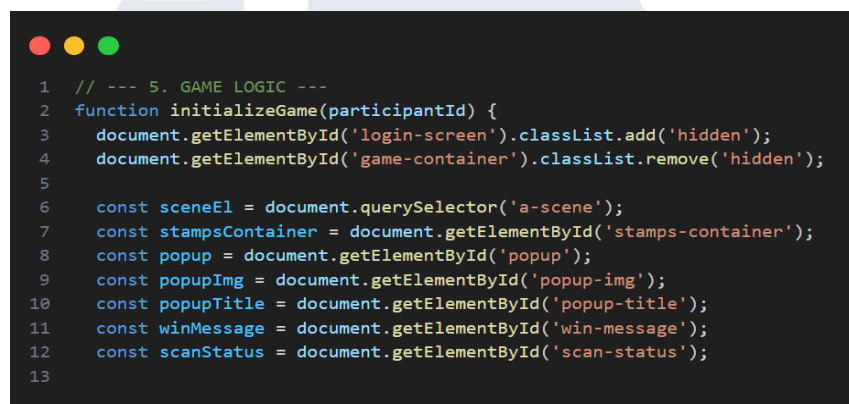
```

Gambar 3.22 Implementasi Fitur Reset Progress Permainan Hunter Game PT XYZ 2

Gambar 3.22 menunjukkan implementasi fitur reset progress yang dikembangkan sebagai mekanisme pemeliharaan aplikasi selama *event* berlangsung. Pada bagian ini, juga ditambahkan *event listener* tersembunyi yang akan menghapus seluruh data peserta dan

progress permainan dari localStorage ketika judul permainan ditekan sebanyak lima kali. Setelah proses reset dilakukan, aplikasi akan dimuat ulang secara otomatis. Fitur ini dirancang untuk mempermudah proses debugging dan mengantisipasi kendala teknis saat aplikasi digunakan oleh banyak peserta dalam kondisi event yang dinamis..

6. Fungsi initializeGame: Mengatur UI, Progress, dan Interaksi Utama



```
1 // --- 5. GAME LOGIC ---
2 function initializeGame(participantId) {
3   document.getElementById('login-screen').classList.add('hidden');
4   document.getElementById('game-container').classList.remove('hidden');
5
6   const sceneEl = document.querySelector('a-scene');
7   const stampsContainer = document.getElementById('stamps-container');
8   const popup = document.getElementById('popup');
9   const popupImg = document.getElementById('popup-img');
10  const popupTitle = document.getElementById('popup-title');
11  const winMessage = document.getElementById('win-message');
12  const scanStatus = document.getElementById('scan-status');
13 }
```

Gambar 3.23 Inisialisasi Logika Utama Permainan Hunter Game (Fungsi initializeGame)

Gambar 3.23 menunjukkan fungsi initializeGame yang dikembangkan sebagai inti dari logika permainan Hunter Game. Pada tahap ini, diatur tampilan antarmuka dengan menyembunyikan halaman login dan menampilkan halaman permainan. Selain itu, fungsi ini memuat kembali data progress gem yang tersimpan pada localStorage dan menyesuaikan indikator visual pada antarmuka. Di dalam fungsi ini, didefinisikan beberapa sub-fungsi utama, seperti showPopup untuk menampilkan notifikasi penemuan gem, updateStampsUI untuk memperbarui progress gems, serta showWinScreen untuk menampilkan halaman kemenangan ketika seluruh gem berhasil dikumpulkan.

7. Inisialisasi MindAR dan Penanganan Event AR (targetFound & targetLost)

```

1 // Init MindAR
2 const targetSrc = new URL('./targets.mind', window.location.href).href;
3 sceneEl.setAttribute('mindar-image', {
4   imageTargetSrc: targetSrc,
5   autoStart: false,
6   uiScanning: '#scan-status',
7   uiLoading: false,
8   uiError: false
9 });
10
11 iconIds.forEach((id, i) => {
12   const entity = document.createElement('a-entity');
13   entity.setAttribute('id', `target-${i}`);
14   entity.setAttribute('mindar-image-target', `targetIndex: ${i}`);
15
16   const iconId = iconIds[i];
17   entity.addEventListener('targetFound', () => handleTargetFound(iconId));
18   entity.addEventListener('targetLost', () => {
19     scanStatus.textContent = "Arahkan kamera ke salah satu gem(s).";
20     scanStatus.classList.remove('text-green-400');
21   });
22
23   sceneEl.appendChild(entity);
24 });

```

Gambar 3.24 Inisialisasi MindAR dan Penanganan Event Target Augmented Reality pada Hunter Game PT XYZ 2

Gambar 3.24 menunjukkan proses inisialisasi modul *MindAR* serta penanganan *event augmented reality* yang dikembangkan. Pada bagian ini, setiap target gambar *AR* dihubungkan dengan indeks tertentu menggunakan file *.mind* sebagai referensi pendeteksian. *Event targetFound* dikaitkan dengan fungsi *handleTargetFound* untuk memproses setiap target yang berhasil dipindai sebagai *gem* yang ditemukan. Sementara itu, *event targetLost* digunakan untuk mengembalikan tampilan instruksi ke kondisi awal. Integrasi ini memungkinkan sistem memproses pemindaian gambar secara *real time* dan menjadi fitur utama dalam mekanisme permainan Hunter Game.

8. Fitur Auto-Start Kamera untuk Kemudahan Pengguna

Pada tahap akhir pengembangan Hunter Game, diimplementasikan fitur *auto-start* kamera untuk meningkatkan kenyamanan pengguna selama event berlangsung. Fitur ini memungkinkan kamera perangkat aktif secara otomatis ketika peserta

membuka kembali aplikasi atau melakukan refresh halaman, tanpa perlu menekan tombol tambahan.

Secara teknis, dirancang fungsi *autoStartCamera* yang akan menunggu hingga seluruh komponen *A-Frame* selesai dimuat sebelum mengaktifkan sistem MindAR. Implementasi ini bertujuan untuk memastikan proses inisialisasi kamera berjalan stabil serta mengurangi potensi kendala teknis, terutama pada kondisi event yang ramai dan melibatkan banyak peserta secara bergantian.

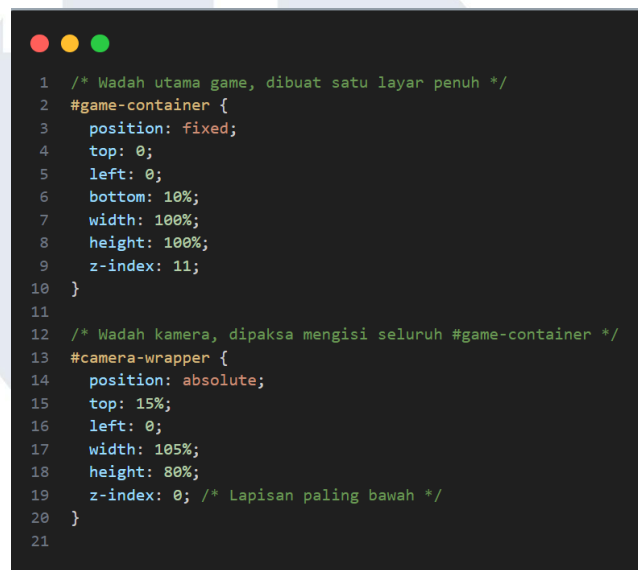
Gambar 3.25 Implementasi Fungsi Auto-Start Kamera pada Aplikasi Hunter Game

Gambar 3.25 menunjukkan implementasi fungsi auto-start kamera yang dikembangkan untuk mengatur mekanisme pemanggilan kamera secara otomatis setelah sistem augmented reality siap digunakan. Dengan adanya fitur ini, alur permainan dapat langsung dimulai tanpa interaksi tambahan dari pengguna, sehingga pengalaman bermain menjadi lebih efisien dan praktis.

Selain pengaturan logika permainan, dilakukan perancangan tampilan antarmuka menggunakan berkas CSS khusus untuk mengatur tata letak kamera dan elemen UI pada Hunter Game. Perancangan ini bertujuan untuk memastikan tampilan kamera augmented reality, area permainan, serta komponen antarmuka dapat ditampilkan secara proporsional dan responsif pada berbagai ukuran

layar, terutama perangkat mobile yang digunakan saat event berlangsung.

Dalam implementasinya, diatur tiga komponen utama, yaitu *#game-container* sebagai wadah utama permainan, *#camera-wrapper* sebagai area tampilan kamera AR, dan *.ui-container* sebagai lapisan antarmuka pengguna yang berada di atas kamera.



```
1  /* Wadah utama game, dibuat satu layar penuh */
2  #game-container {
3    position: fixed;
4    top: 0;
5    left: 0;
6    bottom: 10%;
7    width: 100%;
8    height: 100%;
9    z-index: 11;
10 }
11
12 /* Wadah kamera, dipaksa mengisi seluruh #game-container */
13 #camera-wrapper {
14   position: absolute;
15   top: 15%;
16   left: 0;
17   width: 105%;
18   height: 80%;
19   z-index: 0; /* Lapisan paling bawah */
20 }
21
```

Gambar 3.26 Pengaturan Tata Letak Game Container dan Camera Wrapper pada Hunter Game

Gambar 3.26 menunjukkan pengaturan CSS pada komponen *#game-container* dan *#camera-wrapper* yang telah dirancang. Elemen *#game-container* diatur menggunakan posisi *fixed* untuk menempati seluruh layar, sehingga permainan ditampilkan dalam satu halaman tanpa perpindahan tampilan. Sementara itu, *#camera-wrapper* ditempatkan pada lapisan paling bawah menggunakan posisi *absolute* agar feed kamera MindAR dapat mengisi area permainan secara penuh. Penyesuaian ukuran dan posisi dilakukan untuk memastikan tampilan kamera tetap stabil dan tidak menyisakan ruang kosong pada perangkat tertentu.

```

1  /* Wadah UI, dibuat agar 'mengambang' di atas kamera */
2  .ui-container {
3      position: absolute;
4      top: 0;
5      left: 0;
6      width: 100%;
7      height: 100%;
8      z-index: 10; /* Lapisan di atas kamera */
9      display: flex;
10     flex-direction: column;
11     justify-content: space-between; /* Mendorong header ke atas, footer ke bawah */
12     padding: 2rem;
13     box-sizing: border-box; /* Agar padding tidak merusak ukuran */
14     pointer-events: none; /* Biarkan klik 'menembus' ke kamera */
15 }

```

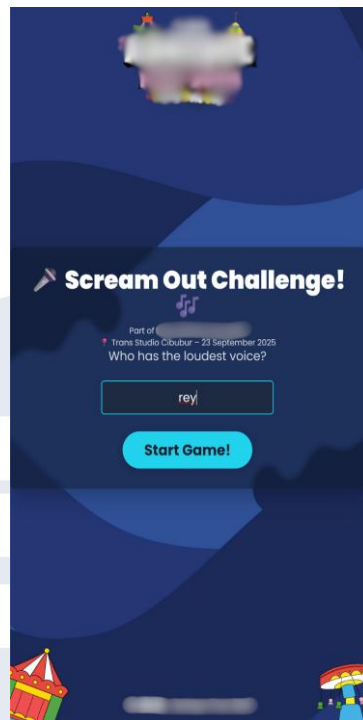
Gambar 3.27 Pengaturan UI Container sebagai Lapisan Antarmuka di Atas Kamera AR

Gambar 3.27 menunjukkan pengaturan CSS pada komponen `.ui-container` yang berfungsi sebagai lapisan antarmuka di atas tampilan kamera. Pada bagian ini, digunakan *flexbox* untuk menyusun elemen header dan footer secara seimbang di bagian atas dan bawah layar. Properti `pointer-events: none` diterapkan agar lapisan antarmuka tidak menghalangi interaksi pengguna dengan kamera di belakangnya. Pengaturan ini memastikan antarmuka tetap informatif tanpa mengganggu proses pemindaian augmented reality selama permainan berlangsung.

3.3.1.3 Scream Wall untuk PT. XYZ 2

Proyek *Scream Out Challenge* merupakan aplikasi permainan interaktif berbasis web yang dikembangkan selama program magang. Aplikasi ini dirancang sebagai media hiburan sekaligus aktivitas interaktif pada kegiatan internal perusahaan, dengan memanfaatkan intensitas suara peserta sebagai dasar penilaian permainan. Tujuan pengembangan proyek ini adalah untuk menciptakan pengalaman permainan yang sederhana, responsif, dan mudah digunakan oleh seluruh peserta acara tanpa memerlukan perangkat tambahan selain mikrofon.

1. Halaman Awal / Input Peserta

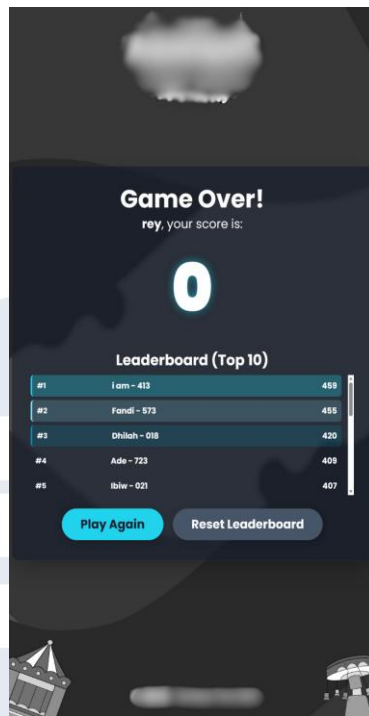


Gambar 3.28 Tampilan Input Nama *Scream Wall Event* PT XYZ 2

Pada gambar 3.28 adalah tahap awal permainan, diimplementasikan halaman input peserta yang berfungsi untuk mengumpulkan nama peserta sebelum permainan dimulai. Halaman ini menampilkan judul permainan “Scream Out Challenge”, deskripsi singkat event, serta sebuah input field untuk memasukkan nama peserta.

Tombol Start Game digunakan untuk memulai permainan sekaligus memicu proses aktivasi microphone pada perangkat peserta. Halaman ini dirancang secara sederhana agar peserta dapat dengan cepat memahami instruksi dan langsung memulai permainan. Implementasi halaman input peserta dapat dilihat pada Gambar 3.28.

2. Halaman Hasil / Game Over



Gambar 3.29 Tampilan Output *Leaderboard Event PT XYZ 2*

Pada gambar 3.29 setelah proses pengukuran suara selesai, aplikasi menampilkan halaman hasil permainan yang berisi skor suara yang berhasil dicapai oleh peserta. Pada halaman ini, sistem menampilkan informasi apakah skor peserta termasuk ke dalam Top 10 leaderboard.

Leaderboard ditampilkan dalam bentuk tabel yang memuat nama peserta dan skor tertinggi yang telah tercatat. Selain itu, dua tombol utama juga diimplementasikan, yaitu *Play Again* untuk mengulang permainan dan *Reset Leaderboard* yang digunakan untuk keperluan pengelolaan data oleh panitia atau admin event.

Tampilan halaman hasil permainan dan leaderboard ini berfungsi sebagai umpan balik visual sekaligus elemen kompetitif yang mendorong partisipasi pengunjung lainnya.

1. Struktur Dasar & Deklarasi Elemen (HTML & DOM Binding)

```

1  const screens = {
2      setup: document.getElementById('setup-screen'),
3      game: document.getElementById('game-screen'),
4      result: document.getElementById('result-screen'),
5  };
6
7  const mainContainer = document.querySelector('.main-container');
8  const startButton = document.getElementById('startButton');
9  const restartButton = document.getElementById('restartButton');
10 const resetLeaderboardButton = document.getElementById('resetLeaderboardButton');
11
12 const playerNameInput = document.getElementById('playerName');
13 const scoreDisplay = document.getElementById('score');
14 const timerDisplay = document.getElementById('timer');
15 const micError = document.getElementById('mic-error');
16 const nameError = document.getElementById('name-error');
17
18 const countdownOverlay = document.getElementById('countdown-overlay');
19 const countdownText = document.getElementById('countdown-text');
20
21 const finalScoreDisplay = document.getElementById('finalScore');
22 const resultPlayerName = document.getElementById('resultPlayerName');
23 const leaderboardDisplay = document.getElementById('leaderboard');
24
25 const initialBackground = document.getElementById('initial-background');
26 const gameBackground = document.getElementById('game-background');
27 const bgGrayscale = document.getElementById('bg-grayscale');

```

Gambar 3.30 Tampilan Struktur Elemen Antarmuka Aplikasi

Pada tahap awal pengembangan aplikasi *Scream Out Challenge*, dibangun struktur dasar antarmuka berbasis HTML yang terdiri dari beberapa komponen utama, yaitu layar persiapan (*setup screen*), layar permainan (*game screen*), dan layar hasil (*result screen*). Selain itu, disediakan elemen pendukung seperti tombol kontrol, input nama peserta, tampilan skor, timer, serta elemen visual latar belakang.

Seperti ditunjukkan pada Gambar 3.30, dilakukan inisialisasi seluruh elemen tersebut menggunakan fungsi `document.getElementById()`. Proses ini bertujuan agar setiap elemen antarmuka dapat dikendalikan secara dinamis melalui JavaScript selama permainan berlangsung.

Selain inisialisasi elemen, beberapa variabel global juga dideklarasikan, seperti skor permainan, sisa waktu, dan status permainan. Variabel-variabel tersebut berfungsi sebagai penyimpanan kondisi permainan secara *real-time* dan menjadi dasar bagi pengolahan logika permainan pada tahap selanjutnya.

2. Logika Pergantian layar



```

1 // Switch screen
2 const switchScreen = (screenName) => {
3   screens[activeScreen].classList.add('hidden');
4   activeScreen = screenName;
5   setTimeout(() => {
6     screens[screenName].classList.remove('hidden');
7   }, 50);
8 };

```

Gambar 3.31 Pergantian Tampilan Antar Layar

Aplikasi dirancang menggunakan konsep beberapa layar (*screen-based interface*) untuk menjaga alur permainan tetap terstruktur dan mudah dipahami oleh pengguna. Terdapat tiga layar utama, yaitu layar pengisian nama peserta, layar permainan, dan layar hasil.

Berdasarkan Gambar 3.31, diimplementasikan fungsi pergantian layar yang bertugas menyembunyikan layar yang tidak aktif dan menampilkan layar yang sesuai dengan tahap permainan. Mekanisme ini digunakan setiap kali pemain berpindah dari tahap persiapan ke permainan, serta dari permainan ke tampilan hasil.

Pendekatan ini bertujuan untuk memisahkan setiap fase permainan secara jelas, sehingga pengguna tidak mengalami kebingungan saat berinteraksi dengan aplikasi.

3. Inisialisasi Game & Akses Mikrofon



```

1 async function initGame() {
2   currentPlayerName = playerNameInput.value.trim();
3   if (!currentPlayerName) {
4     nameError.classList.remove('hidden');
5     return;
6   }
7   nameError.classList.add('hidden');
8   try {
9     const stream = await navigator.mediaDevices.getUserMedia({ audio: true });
10    handleMicSuccess(stream);
11  } catch {
12    micError.classList.remove('hidden');
13  }
14 }
15

```


Gambar 3.32 Proses Inisialisasi Game dan Permintaan Akses Mikrofon

Setelah peserta memasukkan nama dan menekan tombol *Start Game*, sistem menjalankan proses inisialisasi permainan. Pada tahap ini, dilakukan juga validasi input nama serta meminta izin akses mikrofon kepada pengguna.

Seperti ditunjukkan pada Gambar 3.32, akses mikrofon diperoleh melalui pemanggilan API `navigator.mediaDevices.getUserMedia()`. Mikrofon digunakan sebagai sumber input utama untuk mengukur intensitas suara peserta selama permainan berlangsung.

Apabila izin akses diberikan, sistem melanjutkan ke tahap persiapan permainan. Sebaliknya, jika izin ditolak, aplikasi menampilkan pesan kesalahan dan permainan tidak dapat dilanjutkan. Tahap ini menjadi krusial karena keberhasilan permainan sepenuhnya bergantung pada ketersediaan input audio.

4. Persiapan Audio & Analisis Frekuensi (AudioContext +



```
1 function handleMicSuccess(stream) {
2   micError.classList.add('hidden');
3   maintainer.classList.add('game-mode');
4   switchScreen('game');
5
6   initialBackground.style.opacity = '0';
7   gameBackground.style.opacity = '1';
8
9   audioContext = new AudioContext();
10  analyser = audioContext.createAnalyser();
11  analyser.fftSize = 256;
12
13  microphone = audioContext.createMediaStreamSource(stream);
14  microphone.connect(analyser);
15
16  startCountdown();
17 }
18
```

Gambar 3.33 Persiapan Analisis Suara dan Visualisasi Intensitas

Setelah akses mikrofon berhasil diperoleh, proses pengolahan audio disiapkan menggunakan *Web Audio API*. Berdasarkan Gambar

3.33, sistem membangun sebuah *AudioContext* yang dihubungkan dengan *AnalyserNode* untuk membaca data suara secara *real-time*.

Analysers ini memungkinkan sistem memperoleh nilai intensitas suara peserta yang kemudian digunakan sebagai dasar perhitungan skor. Pada tahap ini, antarmuka permainan juga diubah ke mode aktif, ditandai dengan perubahan tampilan latar belakang dari grayscale menjadi berwarna, sebagai indikasi bahwa permainan siap dimulai.

5. Countdown Sebelum Permainan Dimulai



```
1 function startCountdown() {
2   countdownOverlay.classList.remove('hidden');
3
4   let count = 3;
5   countdownText.textContent = count;
6
7   const interval = setInterval(() => {
8     count--;
9
10    if (count > 0) {
11      countdownText.textContent = count;
12    } else if (count === 0) {
13      countdownText.textContent = "SCREAM!";
14    } else {
15      clearInterval(interval);
16      countdownOverlay.classList.add('hidden');
17      startGame();
18    }
19  }, 1000);
20 }
```

Gambar 3.34 Countdown Sebelum Permainan Dimulai

Sebelum permainan dimulai, sistem menjalankan mekanisme hitung mundur selama beberapa detik. Tahap ini bertujuan untuk memberikan waktu persiapan kepada peserta sebelum mulai berteriak.

Berdasarkan Gambar 3.34, hitung mundur ditampilkan dalam bentuk visual overlay pada layar permainan. Pada akhir hitungan, sistem memberikan sinyal visual sebagai tanda bahwa permainan telah dimulai dan proses pengukuran intensitas suara resmi berjalan.

6. Proses Perhitungan Skor Suara (Core Logic)

```

1  function updateGame() {
2    if (!gameRunning) return;
3
4    const dataArray = new Uint8Array(analyser.frequencyBinCount);
5    analyser.getByteFrequencyData(dataArray);
6
7    const average = dataArray.reduce((a, b) => a + b, 0) / dataArray.length;
8    const currentVolumeScore = Math.floor(average * 2.5);
9
10   if (currentVolumeScore > score) {
11     score = currentVolumeScore;
12     scoreDisplay.textContent = score;
13   }
14
15   const targetFill = Math.min(100, (score / 500) * 100);
16   totalFill += (targetFill - totalFill) * 0.05;
17
18   bgGrayscale.style.clipPath = `inset(0 0 ${totalFill}% 0)`;
19
20   requestAnimationFrame(updateGame);
21 }

```

Gambar 3.35 Proses Penghitungan Skor Berdasarkan Intensitas Suara

Selama permainan berlangsung, sistem secara berkelanjutan membaca intensitas suara peserta dan mengonversinya menjadi skor permainan. Skor yang ditampilkan merupakan representasi dari suara paling keras yang dihasilkan peserta selama durasi permainan.

Seperti ditunjukkan pada Gambar 3.35, perubahan intensitas suara juga divisualisasikan melalui perubahan tampilan latar belakang. Visualisasi ini memberikan umpan balik langsung kepada peserta mengenai kekuatan suara yang dihasilkan, sehingga meningkatkan keterlibatan dan pengalaman bermain.

7. Timer 5 Detik & Akhir Permainan

```

1  function startGame() {
2    gameRunning = true;
3    score = 0;
4    timeLeft = 5;
5    totalFill = 0;
6
7    bgGrayscale.style.clipPath = `inset(0 0 0% 0)`;
8
9    scoreDisplay.textContent = 0;
10   timerDisplay.textContent = timeLeft;
11
12   timerInterval = setInterval(() => {
13     timeLeft--;
14     timerDisplay.textContent = timeLeft;
15     if (timeLeft <= 0) endGame();
16   }, 1000);
17
18   updateGame();
19 }

```

Gambar 3.36 Tampilan Skor Akhir dan Pengakhiran Permainan

Permainan dibatasi dalam durasi waktu tertentu. Setelah waktu permainan berakhir, sistem secara otomatis menghentikan proses pengambilan data suara dan menampilkan hasil akhir permainan.

Berdasarkan Gambar 3.36, layar hasil menampilkan nama peserta dan skor yang diperoleh. Tahap ini menandai berakhirnya satu sesi permainan sebelum peserta diarahkan kembali ke tampilan awal atau ke fitur lanjutan.

8. Leaderboard: Penyimpanan, Penyortiran, dan Penampilan Skor

```
1 function displayLeaderboard() {  
2   const scores = getScores();  
3   leaderboardDisplay.innerHTML = '';  
4  
5   if (!scores.length) {  
6     leaderboardDisplay.innerHTML = "<p>No scores yet.</p>";  
7     return;  
8   }  
9  
10  scores.forEach((entry, i) => {  
11    const item = document.createElement('div');  
12  
13    let rankClass = '';  
14    if (i === 0) rankClass = "gold";  
15    else if (i === 1) rankClass = "silver";  
16    else if (i === 2) rankClass = "bronze";  
17  
18    item.className = `leaderboard-item p-3 mb-2 rounded-lg flex justify-between ${rankClass}`;  
19    item.innerHTML = `  
20      <span class="font-bold">#${i + 1}</span>  
21      <span class="font-semibold">${entry.name}</span>  
22      <span class="font-bold">${entry.score}</span>  
23    `;  
24    leaderboardDisplay.appendChild(item);  
25  });  
26 }
```

Gambar 3.37 Leaderboard Skor Tertinggi Peserta

Sebagai fitur pendukung, dikembangkan sistem leaderboard untuk menampilkan daftar skor tertinggi dari seluruh peserta. Data skor disimpan dan dikelola secara lokal, kemudian ditampilkan dalam bentuk peringkat.

Seperti ditunjukkan pada Gambar 3.37, leaderboard menampilkan sepuluh skor tertinggi dengan penanda visual khusus untuk tiga peringkat teratas. Fitur ini bertujuan untuk meningkatkan

aspek kompetitif serta mendorong partisipasi peserta dalam permainan.

9. Reset Permainan & Reset Leaderboard



```
1  }
2
3  function resetLeaderboard() {
4      localStorage.removeItem('screamLeaderboard');
5      displayLeaderboard();
6  }
7
```

Gambar 3.38 Tampilan Fitur Reset Permainan dan Leaderboard

Aplikasi menyediakan dua fitur penting untuk memastikan permainan dapat digunakan secara berulang selama event berlangsung. Fitur pertama adalah reset permainan, yang diaktifkan melalui tombol “*Play Again*”. Fitur ini mengembalikan seluruh parameter permainan ke kondisi awal, termasuk skor, timer, dan tampilan antarmuka, sehingga peserta baru dapat memulai permainan dengan kondisi bersih.

Fitur kedua adalah *reset leaderboard*, yang diaktifkan melalui tombol “*Reset Leaderboard*”. Fitur ini berfungsi untuk menghapus seluruh data skor yang tersimpan, biasanya dilakukan oleh panitia sebelum sesi kompetisi dimulai agar data skor lama tidak mempengaruhi sesi baru. Kedua fitur ini bertujuan untuk menjaga kelancaran jalannya permainan bagi banyak peserta secara berturut-turut.

Seperti ditunjukkan pada Gambar 3.38, tampilan antarmuka reset menampilkan opsi pengembalian kondisi permainan dan penghapusan data skor secara jelas bagi pengguna

3.3.1.4 Wishing Wall untuk event Awarding PT XYZ 3

Proyek keempat yang dikerjakan dalam program magang ini adalah pengembangan aplikasi web interaktif bernama Wishing Wall yang digunakan pada acara Awarding PT XYZ 3. Aplikasi ini dirancang sebagai media digital untuk mengumpulkan dan menampilkan ucapan, harapan, serta pesan apresiasi dari para peserta acara secara real-time pada layar utama event.

Tujuan dari pengembangan Wishing Wall adalah untuk meningkatkan keterlibatan peserta acara melalui interaksi langsung, di mana setiap tamu dapat mengirimkan pesan secara mandiri dan melihat hasilnya ditampilkan secara langsung sebagai bagian dari rangkaian acara.

The image shows a digital interface for a 'Wishing Wall'. At the top, it says 'Wish for 50th anniversary' and 'Indonesia'. Below this is a form with two main sections: 'Nama:' with a text input field containing 'Nama Kamu', and 'Wish:' with a text input field containing 'Tulis Ucapanmu...'. There is a small character count 'Sisa 30 karakter' and a green 'Kirim' button at the bottom of the form. The background of the interface is a green field with a path and stylized white clouds.

Gambar 3.39 Halaman Input Ucapan Peserta Wishing WallEvent PT XYZ 3

Tampilan form input Wishing Wall dapat dilihat pada Gambar 3.39. Halaman ini digunakan oleh peserta acara untuk mengisi **nama** dan pesan ucapan yang ingin disampaikan. Dirancang tampilan form dengan desain sederhana dan fokus agar mudah digunakan oleh berbagai kalangan peserta.

Form diletakkan di atas latar belakang bertema anniversary, dengan elemen card semi-transparan berwarna putih untuk menjaga keterbacaan teks. Setelah tombol “Kirim” ditekan, data ucapan akan diproses oleh sistem dan dikirimkan ke database.



Gambar 3.40 Halaman Output Ucapan Peserta Wishing WallEvent PT XYZ 3

Tampilan output Wishing Wall yang berfungsi sebagai layar display utama dapat dilihat pada Gambar 3.40. Halaman ini digunakan untuk menampilkan seluruh ucapan yang telah dikirimkan oleh peserta dan diproyeksikan ke layar LED selama acara berlangsung.

Setiap ucapan ditampilkan dalam bentuk kartu berwarna emas berbentuk ornamen hexagonal yang memuat nama pengirim dan isi pesan. Layout kartu diatur dalam bentuk *grid* dinamis agar tampilan tetap rapi namun tidak terlihat kaku. Desain visual disesuaikan dengan tema acara yang didominasi warna hijau dan elemen dekoratif putih untuk menjaga konsistensi dengan identitas visual event.

Tampilan ini dirancang responsif sehingga dapat digunakan pada berbagai ukuran layar, mulai dari monitor laptop hingga layar LED berukuran besar.



```

1 <div id="container"></div>
2
3 <script src="https://www.gstatic.com/firebasejs/9.22.0/firebase-app-compat.js"></script>
4 <script src="https://www.gstatic.com/firebasejs/9.22.0/firebase-database-compat.js"></script>
5
6 <script src="script.js"></script>

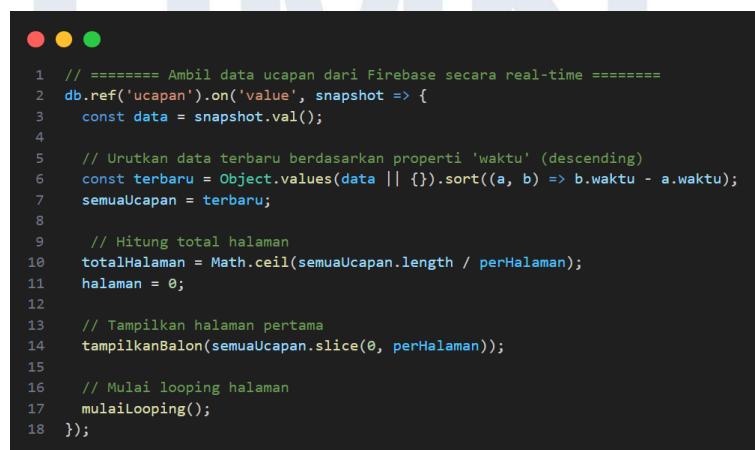
```

Gambar 3.41 Kode Inisialisasi Firebase Wishing Wall

Pada sisi backend, diimplementasikan Firebase Realtime Database sebagai media penyimpanan data ucapan. Proses awal yang dilakukan adalah inisialisasi Firebase menggunakan fungsi `firebase.initializeApp()` yang berisi konfigurasi koneksi aplikasi. Implementasi kode ini ditunjukkan pada Gambar 3.41.

Inisialisasi ini memungkinkan aplikasi untuk terhubung ke node database ucapan, yang menjadi sumber utama data nama dan pesan yang ditampilkan pada layar.

Untuk menampilkan data secara langsung tanpa perlu penyegaran halaman, digunakan listener Firebase dengan fungsi `db.ref('ucapan').on('value', callback)`. Implementasi mekanisme ini dapat dilihat pada Gambar 3.42.



```

1 // ===== Ambil data ucapan dari Firebase secara real-time =====
2 db.ref('ucapan').on('value', snapshot => {
3   const data = snapshot.val();
4
5   // Urutkan data terbaru berdasarkan properti 'waktu' (descending)
6   const terbaru = Object.values(data || {}).sort((a, b) => b.waktu - a.waktu);
7   semuaUcapan = terbaru;
8
9   // Hitung total halaman
10  totalHalaman = Math.ceil(semuaUcapan.length / perHalaman);
11  halaman = 0;
12
13  // Tampilkan halaman pertama
14  tampilkanBalon(semuaUcapan.slice(0, perHalaman));
15
16  // Mulai looping halaman
17  mulaiLooping();
18 });

```

Gambar 3.42 Kode Pengambilan Data Ucapan Real-Time

Setiap kali terdapat ucapan baru yang masuk, sistem secara otomatis mengambil data terbaru, mengubahnya menjadi array, dan

mengurutkannya berdasarkan waktu pengiriman. Ucapan terbaru akan ditampilkan terlebih dahulu pada halaman display.

Fungsi `tampilkanBalon()` juga dikembangkan untuk menampilkan setiap ucapan ke dalam elemen HTML. Fungsi ini bertanggung jawab dalam membentuk kartu ucapan sesuai dengan desain visual acara, seperti yang ditunjukkan pada Gambar 3.43.

```
1 // ===== Fungsi menampilkan balon di layar =====
2 function tampilkanBalon(ucapanList) {
3   const kontainer = document.getElementById('container');
4   kontainer.innerHTML = '';
```

Gambar 3.43 Kode Render Ucapan ke Tampilan Wishing Wall

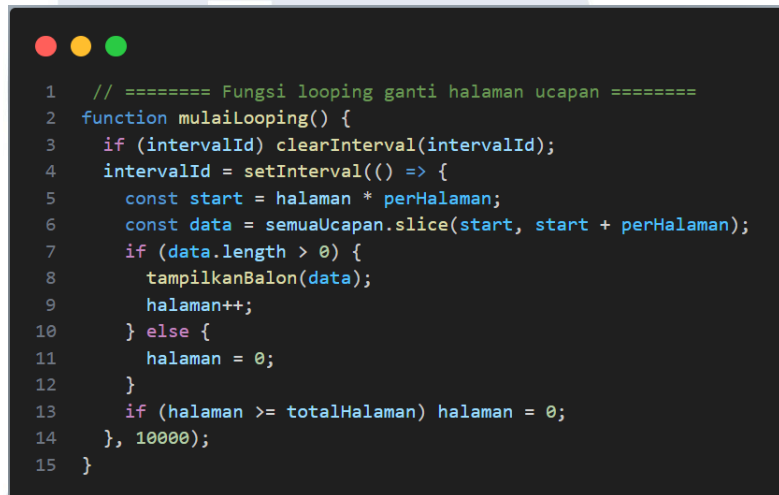
Selain membuat elemen visual, fungsi ini juga menghitung posisi kartu secara dinamis berdasarkan ukuran layar dan margin tertentu agar kartu tidak saling bertumpukan. Pengaturan tata letak dan animasi visual diatur menggunakan CSS, seperti yang ditunjukkan pada Gambar 3.44

```
1 ucapanList.forEach((ucapan, i) => {
2   const div = document.createElement('div');
3   div.className = 'balon';
4
5   // Tentukan posisi kolom dan baris
6   const col = i % cols;
7   const row = Math.floor(i / cols);
8
9   // Hitung posisi (x, y)
10  // Tambah offset untuk geser ke kanan & turun
11  let offsetX = (kontainerWidth - (cols * (balonWidth + marginX) - marginX)) / 2;
12  const offsetY = 60; // geser ke bawah
13
14  // Geser sedikit ke kiri
15  offsetX -= 20; // ubah 60 sesuai kebutuhan (angka lebih besar = lebih ke kiri)
16
17  const x = offsetX + marginX / 2 + col * (balonWidth + marginX);
18  const y = offsetY + marginY + row * (balonHeight + marginY);
19
20  div.style.left = `${x}px`;
21  div.style.top = `${y}px`;
22
23  div.innerHTML = `<div class="isi"><strong>${ucapan.nama}</strong><br>${ucapan.pesan}</div>`;
24  kontainer.appendChild(div);
25  });
26 }
```

Gambar 3.44 Kode Pengaturan Tata Letak Ucapan

Selain itu, sistem juga dilengkapi dengan fungsi `mulaiLooping()` yang bertugas melakukan pergantian halaman secara otomatis setiap 10 detik, kode tersebut dapat dilihat pada gambar 3.45.

Agar seluruh ucapan dapat ditampilkan secara bergantian, diimplementasikan fungsi `mulaiLooping()` yang menggunakan `setInterval()` untuk mengganti halaman tampilan setiap 10 detik. Implementasi mekanisme ini dapat dilihat pada Gambar 3.45.



```
1 // ===== Fungsi looping ganti halaman ucapan =====
2 function mulaiLooping() {
3   if (intervalId) clearInterval(intervalId);
4   intervalId = setInterval(() => {
5     const start = halaman * perHalaman;
6     const data = semuaUcapan.slice(start, start + perHalaman);
7     if (data.length > 0) {
8       tampilkanBalon(data);
9       halaman++;
10    } else {
11      halaman = 0;
12    }
13    if (halaman >= totalHalaman) halaman = 0;
14  }, 10000);
15 }
```

Gambar 3.45 Kode untuk Auto-Looping

Jika halaman terakhir telah ditampilkan, sistem akan kembali ke halaman pertama dan mengulang proses secara terus-menerus. Mekanisme ini memastikan tampilan layar tetap dinamis selama acara berlangsung tanpa perlu pengoperasian manual dari panitia..

3.3.1.5 Quality Assurance firebase html dan Chatbot n8n

Pada proyek kelima, terdapat keterlibatan dalam kegiatan Quality Assurance (QA) untuk sistem berbasis *Firebase*, aplikasi HTML, serta pipeline chatbot yang dibangun menggunakan n8n. Fokus utama dari proyek ini adalah melakukan pengujian fungsional

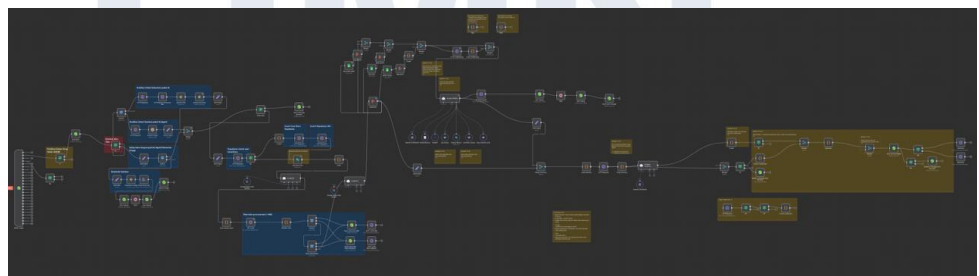
dan stress testing terhadap chatbot sebelum perubahan diterapkan pada sistem utama yang digunakan oleh pengguna.

Dalam proses QA chatbot, digunakan sebuah pipeline khusus untuk pengujian yang diberi nama NADIA. Pipeline ini berfungsi sebagai lingkungan uji terpisah dari pipeline produksi, sehingga seluruh pengujian dapat dilakukan tanpa mengganggu operasional chatbot yang sedang berjalan.

Pipeline NADIA digunakan untuk melakukan berbagai skenario pengujian, seperti:

1. Pengujian alur percakapan chatbot
2. Simulasi pengiriman pesan secara berulang
3. Pengujian ketahanan sistem terhadap beban tinggi (stress testing)

Meskipun pipeline NADIA masih berada dalam tahap pengembangan dan belum sepenuhnya tertata, pipeline ini sudah mencukupi untuk kebutuhan pengujian beban dan deteksi error awal pada node-node chatbot.



Gambar 3.46 *Workflow Pipeline n8n*

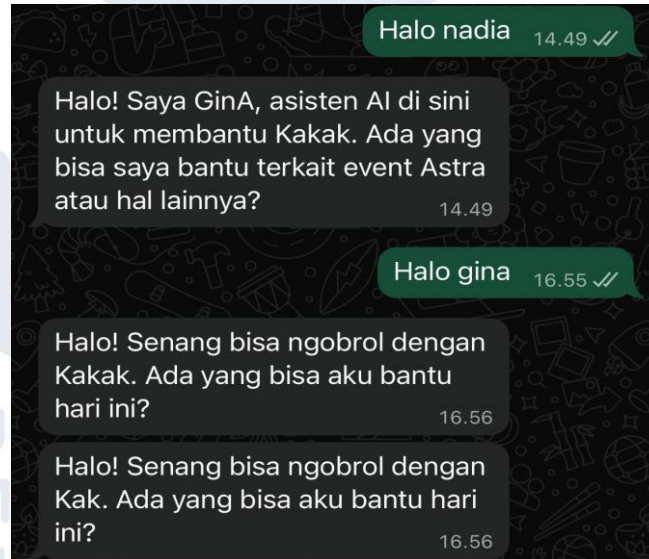
Gambar 3.46 menunjukkan proyek quality assurance untuk pipeline chatbot n8n, digunakan sebuah pipeline khusus untuk pengujian yang dikenal sebagai NADIA. Pipeline ini berfungsi sebagai lingkungan terpisah dari sistem utama, sehingga setiap perubahan atau eksperimen dapat diuji tanpa memengaruhi

operasional chatbot yang sedang berjalan. Dengan adanya pemisahan ini, proses QA dapat dilakukan lebih aman dan terkontrol.

Pipeline NADIA masih berada dalam tahap pengembangan dan belum sepenuhnya tertata rapi, namun sudah memadai untuk kebutuhan stress test [21].

Melalui pipeline ini, dapat diamati perilaku chatbot pada berbagai kondisi uji serta memastikan bahwa setiap perubahan konfigurasi atau logika chatbot berjalan sesuai dengan yang diharapkan sebelum dipindahkan ke pipeline utama.

Salah satu bentuk pengujian yang dilakukan adalah *stress testing*, yaitu pengujian terhadap kemampuan chatbot dalam menangani banyak pesan masuk secara bersamaan. Contoh pelaksanaan stress testing ini ditunjukkan pada Gambar 3.47.



Gambar 3.47 Tampilan stress testing di Whatsapp

Pada pengujian ini, ditemukan menemukan permasalahan berupa pengiriman balasan ganda untuk satu pesan masuk. Masalah ini diduga terjadi akibat tumpang tindih eksekusi node pada pipeline

n8n atau adanya keterlambatan respons dari API WhatsApp yang memicu pengiriman ulang pesan oleh sistem.

Setiap masalah seperti ini akan dilaporkan secara sistematis kepada tim development. Laporan mencakup dokumentasi lengkap, termasuk waktu kejadian, jenis error, log pesan, serta langkah-langkah untuk mereplikasi masalah tersebut. Dengan dokumentasi yang terstruktur, developer dapat melakukan analisis lebih cepat dan menerapkan perbaikan yang tepat pada node terkait agar masalah tidak terulang pada pipeline utama maupun pipeline pengujian.

3.3.1.6 Penginstallan dan implementasi Deepfake live cam AI

Proyek keenam yang dikerjakan adalah penginstallan dan implementasi sistem Deepfake Live Cam AI menggunakan repository Deep-Live-Cam. Proyek ini bertujuan untuk menguji kemampuan sistem dalam melakukan face-swap secara real-time menggunakan input kamera langsung dengan dukungan akselerasi GPU.

Pada tahap perencanaan awal, sistem direncanakan untuk dideploy secara langsung melalui repository GitHub. Namun, dalam proses implementasi, dihadapi beberapa kendala teknis berupa ketidakcocokan versi CUDA dan library onnxruntime dengan spesifikasi hardware server kantor. Ketidaksesuaian ini menyebabkan pipeline inferensi tidak dapat berjalan dengan baik dan beberapa fungsi tidak dapat dieksekusi.

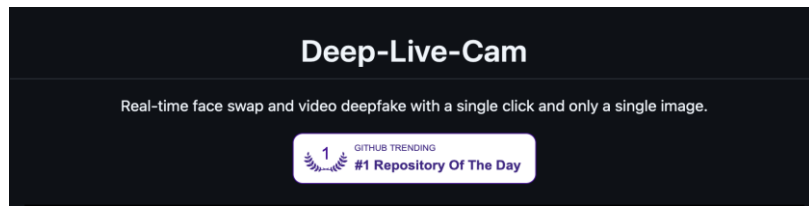
Proyek ini awalnya direncanakan untuk di deploy secara sederhana melalui repository GitHub yang dapat dilihat pada Gambar 3.48, namun implementasinya menjadi jauh lebih kompleks akibat ketidakcocokan antara versi CUDA dan library onnxruntime dengan hardware server kantor. Perbedaan versi ini menyebabkan pipeline inferensi tidak dapat berjalan secara optimal, bahkan beberapa fungsi

tidak dapat dieksekusi sama sekali. Ketidaksesuaian ini berdampak langsung pada proses instalasi, konfigurasi backend, serta pemilihan versi model dan library yang tepat.

Deep-live-cam sendiri merupakan *repository* GitHub yang dirancang untuk melakukan *face-swap* secara langsung dengan memanfaatkan beragam *library computer vision* dan *deep learning*. Sistem ini bekerja dengan memproses input video dari kamera, kemudian melakukan inferensi wajah secara langsung untuk mengganti wajah pengguna dengan wajah target secara instan. Kecepatan dan stabilitas sangat bergantung pada konfigurasi hardware dan kompatibilitas library yang digunakan yang dapat dilihat pada Gambar 3.49.

Salah satu komponen utama dari proyek ini adalah pemanfaatan *torchvision*, yang digunakan untuk melakukan mapping UV pada wajah pengguna. Mapping ini menentukan bagaimana struktur wajah asli diproyeksikan dan disesuaikan dengan wajah target. Proses ini memerlukan performa GPU yang stabil agar perubahan dapat terjadi secara mulus tanpa delay signifikan.

Setelah wajah berhasil diproyeksikan, sistem melakukan blending secara realtime untuk menggabungkan wajah target dengan ekspresi wajah asli pengguna. Tahapan blending ini memerlukan akurasi tinggi agar hasil tampak natural tanpa artefak. Dengan kondisi hardware yang kurang kompatibel, proses blending sering mengalami kegagalan atau lag, sehingga diperlukan solusi alternatif seperti penggunaan *testbench* eksternal atau penyesuaian konfigurasi agar aplikasi dapat berjalan dengan baik.



Gambar 3.48 Repositories Github Deep-Live-Cam

hacksider	Update Quick Start section to v2.3d ✓	dfd145b · 3 weeks ago	🔗 492 Commits
📁 .github/ISSUE_TEMPLATE	Making issue template mandatory	last year	
📁 locales	Add Indonesian localization file	2 months ago	
📁 media	Update Download Button	6 months ago	
📁 models	Added Links to Models in Instructions	last year	
📁 modules	optimization with mac	last month	
📄 .gitattributes	Initial commit	2 years ago	
📄 .gitignore	Update .gitignore	last year	
📄 CONTRIBUTING.md	Added PR checklist	last year	
📄 LICENSE	initial commit	2 years ago	
📄 README.md	Update Quick Start section to v2.3d	3 weeks ago	
📄 mypi.ini	initial commit	2 years ago	
📄 requirements.txt	Add pygrabber to requirements	2 months ago	
📄 run-cuda.bat	Removed default limits	last year	
📄 run-directml.bat	Rename run-laptop-gpu.bat to run-directml.bat	last year	
📄 run.py	Version 2.0c Release!	2 months ago	
📄 tkinter_fix.py	Version 2.0c Release!	2 months ago	

Gambar 3.49 Repositories Github Deep-Live-Cam

Proses instalasi dimulai dengan melakukan cloning repository dari GitHub untuk mendapatkan seluruh struktur proyek dan dependensi dasar yang diperlukan. Setelah repository berhasil diunduh, langkah berikutnya adalah menyiapkan lingkungan kerja terisolasi agar tidak berbenturan dengan konfigurasi Python atau library yang sudah ada di sistem.

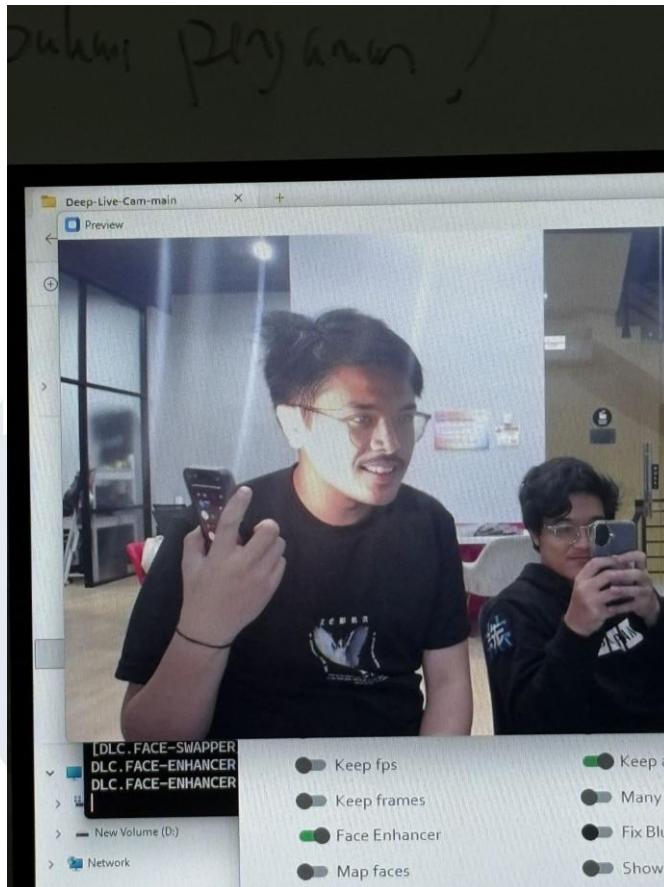
Virtual environment dibuat menggunakan Conda dengan versi Python yang sesuai dengan kebutuhan proyek. Pemilihan versi Python menjadi penting karena beberapa library deep learning memiliki kompatibilitas khusus terhadap versi tertentu. Setelah lingkungan

aktif, instalasi model dilakukan dengan menempatkan file model pada direktori yang telah ditentukan oleh aplikasi.

Selanjutnya, seluruh library yang dibutuhkan diinstal menggunakan file requirement.txt. Proses ini memastikan bahwa versi library yang digunakan sesuai dengan standar proyek, termasuk dependensi untuk computer vision, onnxruntime, dan modul pendukung lainnya. Dependency yang tidak kompatibel akan langsung terdeteksi saat proses instalasi.

Tahap akhir adalah implementasi CUDA pada perangkat GPU untuk mempercepat proses inferensi. Instalasi driver NVIDIA, CUDA Toolkit, dan cuDNN harus sesuai dengan versi yang direkomendasikan oleh library yang digunakan. Ketika seluruh komponen berhasil terpasang, aplikasi dapat dijalankan dengan dukungan akselerasi GPU untuk menghasilkan performa maksimal.





Gambar 3.50 Tampilan Implementasi Deep-Live-Cam

Contoh Deep-Live-Cam yang sudah berjalan menunjukkan kemampuan sistem dalam melakukan face transplan secara real time. Melalui input live feed dari kamera, wajah pengguna dapat dideteksi dengan stabil lalu digantikan dengan wajah target menggunakan model deep learning yang telah dilatih. Proses ini memanfaatkan mekanisme inference yang mampu membaca struktur wajah pengguna dan menyesuaikan titik-titik wajah secara presisi. yang dapat dilihat pada Gambar 3.50.

Setelah wajah target diaplikasikan, sistem melakukan proses blending untuk menyatukan hasil transplan dengan kondisi pencahayaan serta gerakan asli pengguna. Teknik blending ini memastikan pergerakan bibir, ekspresi, dan orientasi kepala tetap

terlihat natural, sehingga hasilnya tampak menyatu dan tidak kaku. Transisi antarframe juga dirender dengan halus untuk menjaga kualitas visual.

Secara keseluruhan, contoh ini memperlihatkan bagaimana teknologi Deep-Live-Cam mampu menghasilkan transformasi wajah secara real time dengan tingkat akurasi tinggi. Sistem ini menciptakan ilusi visual yang realistis berkat kombinasi deteksi wajah yang stabil, face alignment yang presisi, dan blending yang smooth pada output akhir.

3.3.2 Kendala yang Ditemukan

Selama menjalani praktik kerja, dihadapi beberapa kendala yang muncul dalam proses pengembangan proyek, baik yang bersifat teknis maupun terkait proses koordinasi. Tantangan tersebut terutama berkaitan dengan penyesuaian terhadap alur kerja perusahaan, integrasi berbagai tools dan framework, serta perbedaan pemahaman dalam komunikasi tim. Kendala-kendala ini menjadi bagian dari proses pembelajaran dan memberikan gambaran yang lebih jelas mengenai dinamika kerja di lingkungan profesional.

1. Kode yang berantakan dan tidak sinkron akibat miskomunikasi dalam proyek

Kode pada webgame Hunter Game mengalami kekacauan struktur akibat miskomunikasi selama proses pengembangan. Beberapa bagian implementasi tidak mengikuti arahan teknis yang telah ditentukan sebelumnya, sehingga menghasilkan modul dan fungsi yang tidak saling terintegrasi dengan baik. Kondisi ini menyebabkan alur permainan sulit dipelihara dan memunculkan berbagai bug

yang menghambat proses debugging maupun pengembangan fitur lanjutan.

Selain itu, banyak komponen yang dikembangkan tidak sesuai dengan kebutuhan klien. Beberapa fitur dibuat berlebihan atau tidak tepat guna, sementara fitur yang diminta justru tidak terimplementasi secara optimal. Ketidaksesuaian ini membuat workflow permainan menjadi tidak konsisten dan visualisasi tidak mengikuti konsep user experience yang telah disepakati, sehingga memerlukan revisi besar-besaran.

Framework yang digunakan juga tidak diikuti dengan benar, menyebabkan konflik antara struktur proyek dan library yang seharusnya menjadi fondasi pengembangan. Banyak fungsi Javascript tidak kompatibel dengan gaya penulisan dan mekanisme rendering framework, sehingga memperburuk kestabilan program. Akibatnya, sebagian besar kode perlu dirapikan, disesuaikan ulang, dan diselaraskan dengan standar framework agar pengembangan berikutnya dapat berjalan lebih efisien.

2. Permasalahan di backend webgame hunter game bertabrakan dengan framework tailwinds

Javascript pada webgame Hunter Game mengalami kendala saat meminta akses kamera, karena proses permintaan tersebut bertabrakan dengan mekanisme rendering Tailwind CSS. Framework ini melakukan proses re-render elemen secara dinamis, yang menyebabkan inisialisasi kamera dipicu lebih dari satu kali. Akibatnya, browser menolak permintaan akses kamera berikutnya karena permintaan sebelumnya belum dilepas atau masih dianggap aktif.

Masalah ini membuat fitur kamera tidak stabil dan sering gagal dimuat, meskipun logika Javascript sebenarnya sudah benar. Ketidaksinkronan antara lifecycle Tailwind dan proses inisialisasi kamera menciptakan kondisi konflik, terutama ketika component-style Tailwind memicu update visual yang tidak diantisipasi oleh script.

3. Javascript di proyek wishing wall bertabrakan dengan css

Javascript pada proyek Wishing Wall tidak dapat menjalankan fungsi untuk mengubah penataan elemen maupun memicu animasi karena struktur CSS yang masih berantakan dan tidak konsisten. Beberapa class dan styling tidak tertata dengan baik, sehingga selector pada Javascript gagal menemukan elemen target atau tidak dapat menerapkan perubahan gaya secara dinamis. Akibatnya, animasi yang seharusnya muncul secara interaktif tidak berjalan dan tampilan tidak berubah sesuai logika program.

Masalah ini juga disebabkan oleh konflik antar-class, penggunaan CSS yang override satu sama lain, serta tidak adanya standar penamaan yang rapi. Ketika CSS tidak terstruktur, perubahan visual yang dikendalikan oleh Javascript tidak dapat dieksekusi dengan benar, sehingga keseluruhan interaksi menjadi tidak responsif. Penataan ulang CSS diperlukan agar setiap elemen memiliki hierarki dan struktur yang jelas sehingga dapat diakses dan dimodifikasi oleh Javascript tanpa hambatan.

4. Miskomunikasi pada saat quality assurance pada saat testing chatbot ada perbedaan antara pipeline testing dan pipeline live

Terdapat perbedaan mendasar antara pipeline n8n pada environment live dan environment testing, terutama dalam cara sistem mengelola API WhatsApp. Perbedaan tersebut mencakup penanganan nomor telepon, UID, dan JID yang tidak sepenuhnya seragam pada kedua environment. Akibatnya, beberapa fungsi yang bekerja dengan baik di pipeline testing tidak selalu menghasilkan output yang sama ketika dijalankan di pipeline live.

Selain itu, pipeline n8n pada environment live terkadang tidak menerima pembaruan terbaru secara langsung, baik karena perbedaan konfigurasi deployment maupun karena cache yang masih menyimpan versi lama dari node atau parameter. Ketidaksinkronan ini menyebabkan hasil eksekusi workflow berbeda antara keduanya, sehingga debugging menjadi lebih kompleks.

Perbedaan dalam proses deployment juga menuntut pelaksanaan QA secara menyeluruh sebelum pipeline dipindahkan dari testing ke live. Setiap perubahan kecil pada node, parameter, atau API handling perlu diuji secara detail untuk memastikan tidak terjadi error pada operasional harian, terutama karena sistem berinteraksi langsung dengan pengguna melalui WhatsApp. Proses QA yang ketat menjadi kunci untuk menjaga kestabilan dan konsistensi pipeline di lingkungan produksi.

5. Masalah ketidakcocokan NVIDIA CUDA dengan libraries onnxruntime

Hardware server kantor yang digunakan untuk menjalankan *Deep Livecam* tidak sepenuhnya kompatibel dengan library yang dibutuhkan oleh aplikasi.

Ketidaksesuaian ini terutama disebabkan oleh versi CUDA pada server yang terlalu baru sehingga tidak dapat digunakan oleh onnxruntime versi yang menjadi fondasi utama dari sistem Deep Live Cam. Kondisi ini menyebabkan modul-komponen yang bergantung pada kompatibilitas CUDA gagal berfungsi sebagaimana mestinya, terutama pada proses inferensi real-time.

Selain itu, penggunaan GPU NVIDIA RTX 40 Series juga memunculkan kendala baru karena beberapa fitur onnxruntime versi lama tidak lagi didukung pada arsitektur GPU generasi terbaru. Ketidaksesuaian fitur ini menghambat eksekusi pipeline pemrosesan video, yang sangat mengandalkan interoperabilitas antara driver GPU, CUDA, dan modul onnxruntime. Alhasil, beberapa tahap pemrosesan tidak dapat berjalan stabil dan mengakibatkan kegagalan pada proses masking.

Masalah utama yang muncul adalah masking fail, yaitu kondisi ketika sistem gagal menerapkan pemisahan foreground-background secara akurat. Masking yang seharusnya dilakukan oleh model AI tidak berjalan karena modul pemrosesan grafis tidak dapat dipanggil oleh onnxruntime. Akibatnya, output video menjadi tidak layak digunakan dalam konteks aplikasi yang membutuhkan performa real-time dan akurasi visual.

Kendala perangkat keras dan ketidaksesuaian library ini menunjukkan bahwa upgrade library tidak dapat dilakukan secara sembarangan. Versi onnxruntime yang lebih baru memang dapat mendukung GPU generasi baru, tetapi tidak sepenuhnya kompatibel dengan pipeline *Deep Livecam* yang

sebelumnya dibangun dengan dependensi tetap. Penyesuaian harus dilakukan secara menyeluruh agar sistem dapat berjalan stabil pada kombinasi hardware–software yang baru.

Solusi yang memungkinkan adalah melakukan rework pada deployment environment, baik dengan menyesuaikan versi CUDA dan driver GPU, melakukan migrasi library ke versi terbaru, atau memodifikasi pipeline *Deep LiveCam* agar kompatibel dengan environment baru. Pendekatan ini memerlukan pengujian menyeluruh untuk memastikan semua modul tetap berfungsi tanpa memunculkan error baru.

3.3.3 Solusi atas Kendala yang Ditemukan

Bagian ini berisi solusi atas kendala/kesulitan yang ditemukan selama menjalani praktek kerja.

1. Kode harus di perbaiki dan dibuat lebih rapih agar mempermudah pengerjaan

Kode diorganisasi ulang dengan memisahkan setiap fungsi ke dalam bagian yang lebih terstruktur. Pemisahan ini dilakukan untuk meningkatkan keterbacaan, mempermudah proses debugging, serta memastikan setiap modul memiliki tanggung jawab yang jelas. Dengan struktur yang lebih teratur, pengembangan lanjutan dapat dilakukan tanpa menimbulkan konflik antarbagian.

Perubahan mendasar juga diterapkan pada layout dan alur kerja webgame agar lebih sesuai dengan kebutuhan klien. Penyesuaian dilakukan pada elemen tampilan, responsivitas, serta mekanisme interaksi dalam game sehingga pengalaman pengguna dapat mengikuti konsep yang telah disepakati.

Pembaruan ini mencakup perbaikan tata letak dan kontrol visual agar operasional game lebih intuitif.

Selain itu, sistem penyimpanan yang sebelumnya menggunakan cache berbasis JavaScript diganti menjadi penyimpanan menggunakan Firebase. Perubahan ini membuat data lebih aman, tersinkronisasi, dan dapat diakses secara real time. Siklus kerja game (game lifecycle) juga diselaraskan kembali melalui JavaScript agar transisi antarstate seperti start, gameplay, dan game over berjalan lebih lancar dan konsisten.

2. Tailwind css di nonaktifkan karena dinilai tidak terlalu membantu dengan adanya css dan javascript yang lebih sesuai dengan permintaan klien

Penggunaan Tailwind CSS tidak diterapkan sepenuhnya karena struktur proyek menuntut fleksibilitas visual yang lebih spesifik sesuai tema yang ditetapkan klien. Untuk memenuhi kebutuhan tersebut, styling diarahkan menggunakan CSS murni agar tampilan dapat meniru konsep desain secara presisi tanpa batasan utility class. Dengan pendekatan ini, penyesuaian detail seperti warna, posisi elemen, dan estetika keseluruhan dapat dilakukan lebih bebas dan tepat sasaran.

Sementara itu, JavaScript memegang peran utama dalam mengelola animasi, backend database, serta environment game. Pengalihan fungsi ini juga dilakukan untuk menghindari konflik permintaan akses kamera yang terjadi ketika Tailwind dan JavaScript menjalankan request bersamaan. Dengan mengendalikan seluruh mekanisme logika melalui JavaScript, sistem menjadi lebih stabil, tidak terjadi

double access request, dan game dapat berjalan dengan konsisten pada berbagai skenario penggunaan.

3. CSS dibereskan terlebih dahulu dan positioning di serahkan ke javascript

Perombakan menyeluruh dilakukan pada struktur CSS dengan menghapus pemetaan tetap yang sebelumnya tertanam pada beberapa class dan div. Penghapusan ini bertujuan menghilangkan batasan tata letak sehingga elemen-elemen utama tidak lagi terikat pada struktur statis yang menghambat fleksibilitas perubahan. Dengan pendekatan baru, tampilan menjadi lebih mudah diatur ulang tanpa menimbulkan konflik gaya.

Setelah pemetaan CSS dihapus, penataan posisi elemen sepenuhnya dialihkan ke kontrol melalui JavaScript. Dengan cara ini, lokasi setiap elemen dapat diatur secara dinamis sesuai kebutuhan, termasuk jumlah komponen atau bentuk interaksi yang diinginkan klien. Fleksibilitas ini memungkinkan tampilan disesuaikan dengan cepat untuk berbagai versi desain maupun perubahan konsep yang mendadak.

Selain itu, animasi tambahan diterapkan untuk meningkatkan daya tarik visual dan interaksi pengguna. Implementasi animasi dilakukan bersamaan dengan logika JavaScript sehingga transisi dan pergerakan elemen dapat menyesuaikan kondisi real-time di layar. Pembaruan ini tidak hanya membuat tampilan lebih hidup, tetapi juga memastikan pengalaman pengguna menjadi lebih menarik dan responsif.

4. Testing dilakukan secara paralel dengan koordinasi dari development agar dapat mengetes bagian yang penting dan sudah di deploy saja

Pengujian chatbot akhirnya dilakukan secara terarah setelah seluruh node pada pipeline testing berhasil diselesaikan. Pengujian ini difokuskan pada alur fungsional utama untuk memastikan setiap node berjalan sesuai rancangan tanpa mengalami error sistemik. Dengan pendekatan ini, proses validasi dapat dilakukan secara lebih efisien karena setiap perubahan dapat langsung diuji pada bagian yang relevan.

Stress testing tidak dilakukan secara menyeluruh karena adanya perbedaan mendasar antara environment pipeline testing dan pipeline live. Perbedaan tersebut meliputi struktur data WhatsApp API, parameter JID dan UID, serta cara masing-masing pipeline menangani konteks dan routing. Kondisi ini membuat simulasi beban besar di lingkungan testing menjadi tidak akurat dan berpotensi memberikan hasil yang menyesatkan.

Sebagai gantinya, stress testing difokuskan hanya pada masalah-masalah spesifik yang berpotensi muncul pada pipeline live, seperti duplikasi pesan, delay respons, dan ketahanan nodes terhadap input berulang. Pengujian terarah ini dilakukan untuk memastikan bahwa fitur kritis seperti anti-spam, update summary, dan RAG bekerja stabil ketika di-deploy. Pendekatan ini dianggap paling efektif mengingat keterbatasan environment dan perbedaan deployment antara kedua pipeline.

5. CPU offloading untuk testing dan digunakanya hardware yang memadai requirements

Pengujian awal aplikasi dilakukan dengan menggunakan komputasi berbasis CPU karena ketidakcocokan antara perangkat keras server kantor dan kebutuhan library yang digunakan oleh proyek *Deep Live Cam*. Penggunaan CPU ini menyebabkan performa aplikasi menurun signifikan, terutama pada bagian pemrosesan video yang hanya mampu menghasilkan 2–3 fps. Frame rate yang sangat rendah ini berdampak langsung pada ketidaknyamanan penggunaan dan tidak memenuhi standar minimal yang diperlukan untuk pengujian fitur masking secara real time.

Upaya untuk memaksa penggunaan CUDA telah dicoba dengan mengganti dan menyesuaikan versi *onnxruntime* serta *torch*, namun seluruh percobaan tetap mengalami kegagalan. Penyebab utama berasal dari versi CUDA yang terlalu baru serta ketidakcocokan arsitektur GPU NVIDIA RTX 40 Series dengan versi *onnxruntime* yang diperlukan oleh aplikasi. Ketidaksesuaian ini membuat backend GPU tidak dapat berjalan stabil dan tetap kembali pada fallback CPU.

Selain itu, beberapa library inti yang digunakan dalam proyek *Deep Livecam* tidak mendukung driver terbaru yang ada pada server kantor, sehingga memicu error pada proses inisialisasi GPU. Error tersebut mengakibatkan aplikasi tidak dapat melakukan *hardware acceleration*, yang seharusnya menjadi komponen utama untuk menjalankan masking dan pemrosesan video secara optimal. Kondisi ini memperjelas

bahwa terdapat gap teknis antara kebutuhan aplikasi dan spesifikasi perangkat keras yang tersedia.

Keterbatasan CPU dalam melakukan *real-time inference* juga memberikan hambatan besar dalam melakukan iterasi pengujian. Dengan tingkat fps yang rendah, setiap perubahan yang dilakukan pada pipeline aplikasi membutuhkan waktu lebih lama untuk divalidasi. Hal ini memperlambat proses debugging dan berdampak pada mundurnya timeline pengembangan. Situasi tersebut kurang ideal untuk kebutuhan proyek yang sangat bergantung pada respon visual cepat.

Setelah mempertimbangkan berbagai kemungkinan solusi, diputuskan bahwa penggunaan komputer testbench lain yang memiliki spesifikasi GPU lebih mendekati kebutuhan *Deep Live Cam* merupakan opsi paling efektif. Perangkat tersebut dilengkapi GPU yang lebih kompatibel dengan versi CUDA dan *onnxruntime* yang digunakan, sehingga berpotensi mengatasi hambatan teknis yang terjadi di server utama kantor. Pengujian kemudian diarahkan untuk dilakukan melalui testbench ini.

Dengan keputusan ini, proses pengembangan dapat kembali berjalan lebih efisien karena lingkungan pengujian menjadi lebih stabil dan sesuai dengan kebutuhan teknis aplikasi. Penggunaan perangkat yang kompatibel memungkinkan *hardware acceleration* berfungsi optimal, sehingga proses masking dan inferensi video dapat dilakukan dengan frame rate yang lebih tinggi dan mendukung *real-time performance* sesuai tujuan awal proyek.