LAMPIRAN

Lampiran 01. Instrumen Uji ahli isi

PENGEMBANGAN GAME EDUKASI UNTUK PENGENALAN MUSIK TRADISIONAL DENGAN DUKUNGAN VALIDASI SOAL BERBASIS CONVLSTM

Nama Penguji : .....

Profesi/Keahlian : .....

Hari, Tanggal : .....

A. Petunjuk Pengisian:

Berilah tanda centang (√) pada kolom yang tersedia sesuai dengan penilaian.

B. Instrumen Penilaian

No	Aspek/Kriteria Uji	Sesuai	Tidak Sesuai	Catatan
I.	**Validitas Konten**			
1	Kebenaran dan akurasi soal sesuai dengan materi yang disajikan			
2	Aplikasi memberikan umpan balik yang sesuai (misalnya, umpan balik jawaban benar/salah untuk game).			
3	Materi kuis mendukung kompetensi pembelajaran budaya atau musik tradisional.			
4	Penggunaan teks, gambar, dan audio disajikan secara proporsional dan efektif.			
II.	**Relevansi**			
5	Soal kuis sesuai dengan karakteristik dan kebutuhan target pengguna.			
6	Tingkat kesulitan soal atau tantangan disajikan secara bertahap (mulai dari yang mudah ke sulit).			

No	Aspek/Kriteria Uji	Sesuai	Tidak Sesuai	Catatan
7	Konten kuis mampu mendorong minat pengguna untuk mengenal musik tradisional.			
III.	**Kejelasan Isi**			
8	Soal disajikan dengan bahasa yang jelas, ringkas, dan mudah dipahami oleh pengguna.			
9	Keterbacaan teks dan kejelasan penyajian informasi melalui tata letak, grafis, tombol, dan ikon.			
10	Struktur penyajian materi yang logis, sistematis, dan runtut sehingga mudah diikuti alurnya.			

C. Saran

……

……

……

D. Kesimpulan:

Lingkari salah satu opsi di bawah ini!

1. Valid dan dapat digunakan tanpa revisi.
2. Perlu revisi minor sebelum digunakan.
3. Perlu revisi besar sebelum layak digunakan.

Singaraja,
Ahli Isi,

(……...)

Lampiran 02. Instrumen Uji Respon Pengguna

Uji Respon Pengguna
Pengembangan *Game* Harmoni Nusantara untuk Pengenalan Musik Tradisional Nusantara Berbasis ConvLSTM

Identitas Responden:

Nama :

Usia :

Pekerjaan :

Hari, Tanggal :

A. Petunjuk Pengisian:

Nilailah game ini dengan mencentang (√) satu lingkaran di setiap baris. Pilihan Anda menunjukkan kedekatan dengan kata di sisi kiri atau kanan.

B. Instrumen Penilaian Aplikasi Game

Membosankan	O	O	O	O	O	O	O	Mengasyikkan
Rumit	O	O	O	O	O	O	O	Sederhana
Tidak Menarik	O	O	O	O	O	O	O	Menarik
Mengganggu	O	O	O	O	O	O	O	Mendukung
Biasa Saja	O	O	O	O	O	O	O	Kreatif
Tidak Efisien	O	O	O	O	O	O	O	Efisien
Ketinggalan Zaman	O	O	O	O	O	O	O	Modern
Membingungkan	O	O	O	O	O	O	O	Jelas

C. Saran :

.....

.....

.....

Singaraja,
Pengguna,

(……...)

Lampiran 03. Instrumen Uji Media

Uji Media
Pengembangan *Game* Harmoni Nusantara untuk Pengenalan Musik Tradisional Nusantara Berbasis ConvLSTM

Identitas Responden:

Nama :

Jabatan :

Institusi : **Dago Engineering**

Hari, Tanggal :

A. Petunjuk Pengisian:

Mohon Bapak/Ibu untuk memberikan penilaian terhadap media yang dikembangkan dengan memberikan tanda centang (✓) pada kolom skor yang paling sesuai (1 Sangat Tidak Sesuai, 2 Tidak Sesuai, 3 Cukup Sesuai, 4, Sesuai, 5 Sangat Sesuai). Bapak/Ibu juga dapat memberikan catatan, kritik, atau saran pada kolom yang telah disediakan sebagai bahan perbaikan dan penyempurnaan media.

B. Instrumen Penilaian Aplikasi Game

No	Aspek Penilaian	Skor				
		1	2	3	4	5
I.	**Efektivitas (Mekanika & Logika Game)**					
1	Aturan dan tujuan permainan jelas serta mudah dipahami oleh pemain.					
2	Logika permainan (misal: skor, progres, kondisi menang/kalah) berjalan sesuai rencana.					
3	Fungsionalitas inti (core gameplay) berjalan lancar tanpa *bug*.					
II.	**Efisiensi (Pengalaman Pengguna & Kontrol)**					
4	Sistem kontrol (misal: tombol, sentuhan) responsif dan mudah digunakan.					
5	Navigasi menu (menu utama, pengaturan, dll.) mudah dioperasikan.					
6	Umpan balik (*feedback*) dari sistem (misal: UI dari jawaban benar) jelas dan membantu.					
III.	**Aspek Estetika & Kepuasan**					
7	Desain karakter dan lingkungan (grafis) menarik dan konsisten secara visual.					
8	Musik latar dan efek suara (audio) sesuai dengan tema dan suasana permainan.					

C. Instrumen Penilaian Aplikasi Web

No	Aspek Penilaian	Skor				
		1	2	3	4	5
I.	**Efektivitas (Fungsionalitas)**					
1	Keberhasilan sistem mengklasifikasikan audio yang diunggah.					
2	Keberhasilan sistem menghasilkan soal dalam format JSON yang valid.					
3	Alur kerja *end-to-end* (dari unggah hingga unduh JSON) berjalan lancar tanpa eror.					
II.	**Efisiensi (Kemudahan Pengguna)**					
4	Langkah-langkah untuk menggunakan fitur jelas dan tidak berbelit-belit.					
5	Antarmuka pengguna (UI) mudah dipahami dan intuitif.					
6	Informasi dan instruksi yang disajikan pada web mudah dimengerti.					
III.	**Tampilan (Desain Visual & Kepuasan)**					
7	Tata letak (layout) elemen pada halaman terstruktur dengan baik.					

D. Saran:

.....

.....

.....

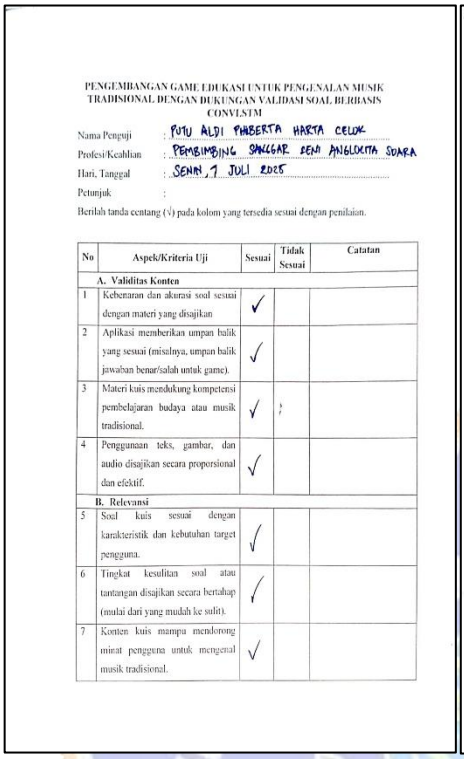
.....

Singaraja,

Ahli,

(……...)

Lampiran 0.4 Hasil Uji Ahli Isi

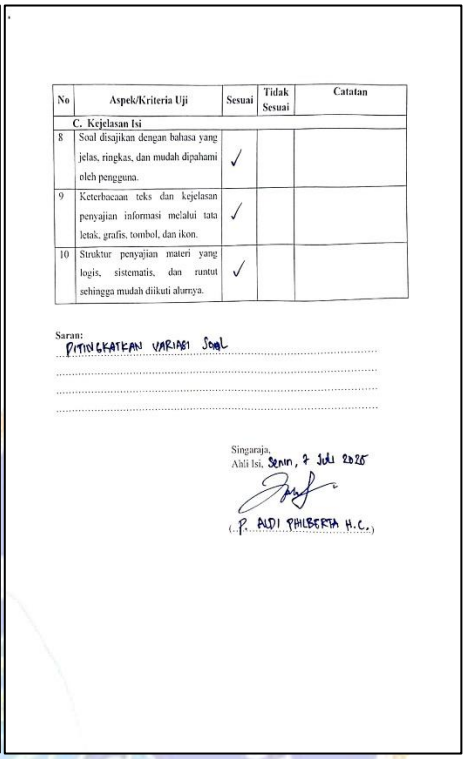


PENGEMBANGAN GAME EDUKASI UNTUK PENGENALAN MUSIK TRADISIONAL DENGAN DUKUNGAN VALIDASI SOAL BERBASIS CONVLSTM

Nama Penguji : PUTU ALDI PHIBERTA HARTA CELUK
Profesi/Keahlian : PEMBIMBING SANGGAR SENI ANGLOKITA SUARA
Hari, Tanggal : SENIN, 7 JULI 2025
Petunjuk :

Berilah tanda centang (√) pada kolom yang tersedia sesuai dengan penilaian.

No	Aspek/Kriteria Uji	Sesuai	Tidak Sesuai	Catatan
	A. Validitas Konten			
1	Kebenaran dan akurasi soal sesuai dengan materi yang disajikan	√		
2	Aplikasi memberikan umpan balik yang sesuai (misalnya, umpan balik jawaban benar/salah untuk game).	√		
3	Materi kuis mendukung kompetensi pembelajaran budaya atau musik tradisional.	√		
4	Penggunaan teks, gambar, dan audio disajikan secara proporsional dan efektif.	√		
	B. Relevansi			
5	Soal kuis sesuai dengan karakteristik dan kebutuhan target pengguna.	√		
6	Tingkat kesulitan soal atau tantangan disajikan secara bertahap (mulai dari yang mudah ke sulit).	√		
7	Konten kuis mampu mendorong minat pengguna untuk mengenal musik tradisional.	√		



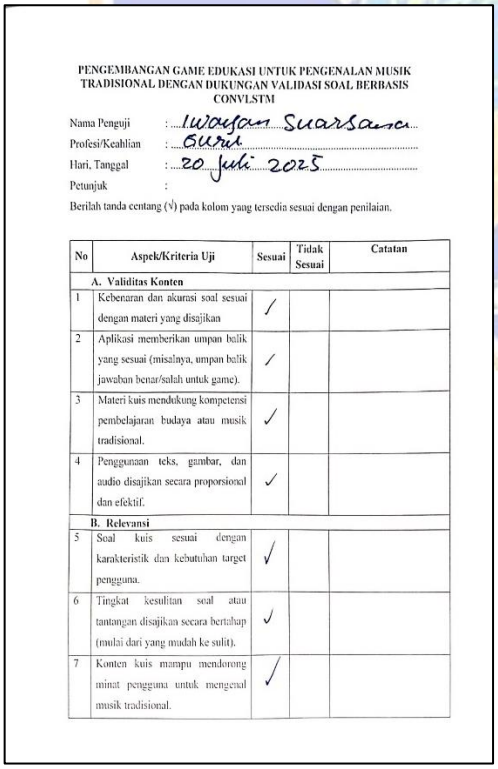
No	Aspek/Kriteria Uji	Sesuai	Tidak Sesuai	Catatan
	C. Kejelasan Isi			
8	Soal disajikan dengan bahasa yang jelas, ringkas, dan mudah dipahami oleh pengguna.	√		
9	Keterbacaan teks dan kejelasan penyajian informasi melalui tata letak, grafis, tombol, dan ikon.	√		
10	Struktur penyajian materi yang logis, sistematis, dan runtut sehingga mudah diikuti alurnya.	√		

Saran: PENINGKATKAN VARIASI SOAL

Singaraja, Senin, 7 Juli 2025
Ahli Isi,

(P. ALDI PHIBERTA H.C.)

Instrumen Penguji 1

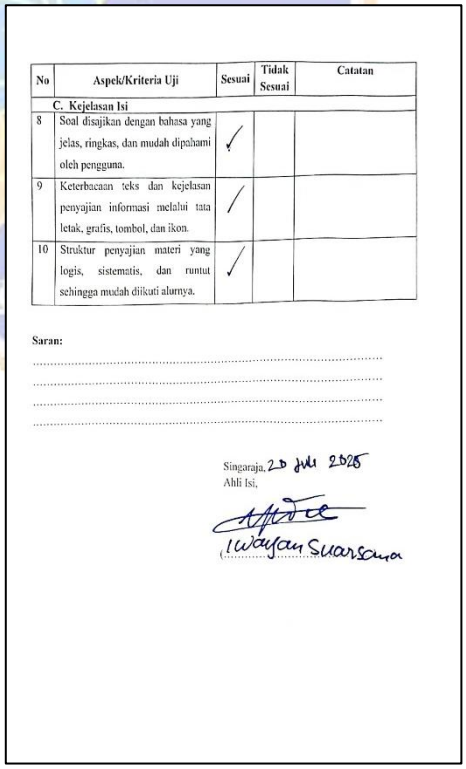


PENGEMBANGAN GAME EDUKASI UNTUK PENGENALAN MUSIK TRADISIONAL DENGAN DUKUNGAN VALIDASI SOAL BERBASIS CONVLSTM

Nama Penguji : I Wayan Suarsana
Profesi/Keahlian : Guru
Hari, Tanggal : 20 Juli 2025
Petunjuk :

Berilah tanda centang (√) pada kolom yang tersedia sesuai dengan penilaian.

No	Aspek/Kriteria Uji	Sesuai	Tidak Sesuai	Catatan
	A. Validitas Konten			
1	Kebenaran dan akurasi soal sesuai dengan materi yang disajikan	√		
2	Aplikasi memberikan umpan balik yang sesuai (misalnya, umpan balik jawaban benar/salah untuk game).	√		
3	Materi kuis mendukung kompetensi pembelajaran budaya atau musik tradisional.	√		
4	Penggunaan teks, gambar, dan audio disajikan secara proporsional dan efektif.	√		
	B. Relevansi			
5	Soal kuis sesuai dengan karakteristik dan kebutuhan target pengguna.	√		
6	Tingkat kesulitan soal atau tantangan disajikan secara bertahap (mulai dari yang mudah ke sulit).	√		
7	Konten kuis mampu mendorong minat pengguna untuk mengenal musik tradisional.	√		



No	Aspek/Kriteria Uji	Sesuai	Tidak Sesuai	Catatan
	C. Kejelasan Isi			
8	Soal disajikan dengan bahasa yang jelas, ringkas, dan mudah dipahami oleh pengguna.	√		
9	Keterbacaan teks dan kejelasan penyajian informasi melalui tata letak, grafis, tombol, dan ikon.	√		
10	Struktur penyajian materi yang logis, sistematis, dan runtut sehingga mudah diikuti alurnya.	√		

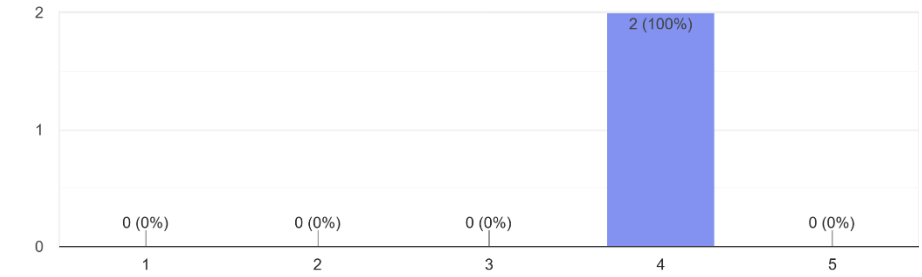
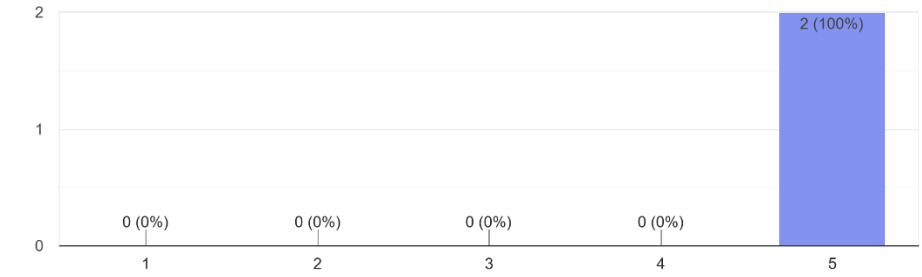
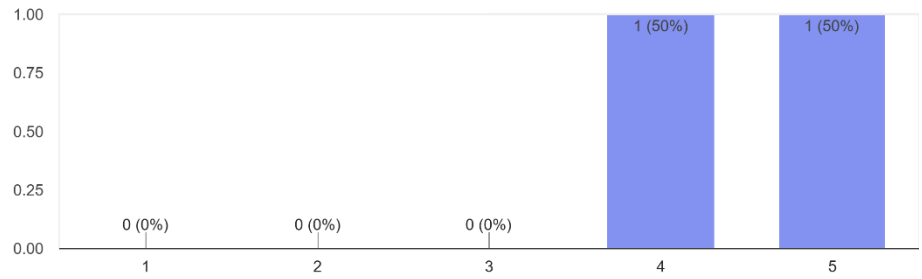
Saran:

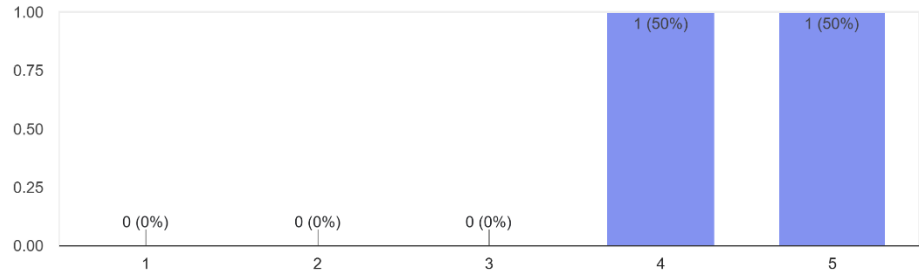
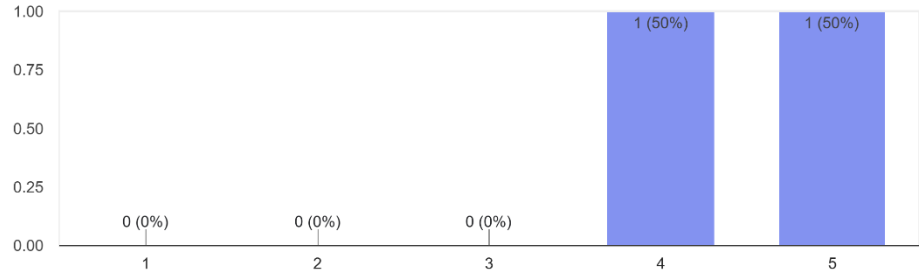
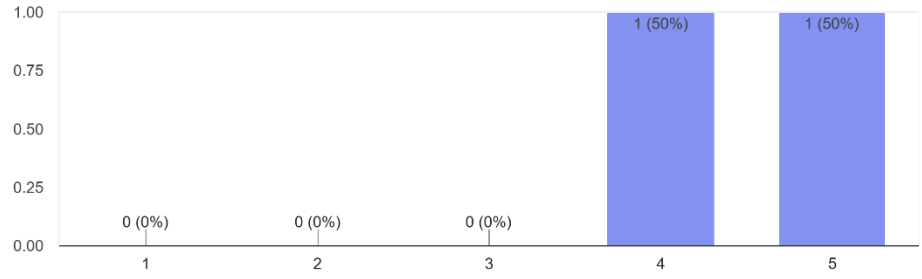
Singaraja, 20 Juli 2025
Ahli Isi,

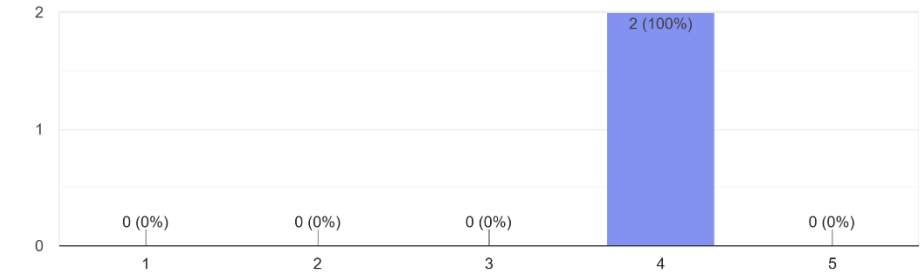
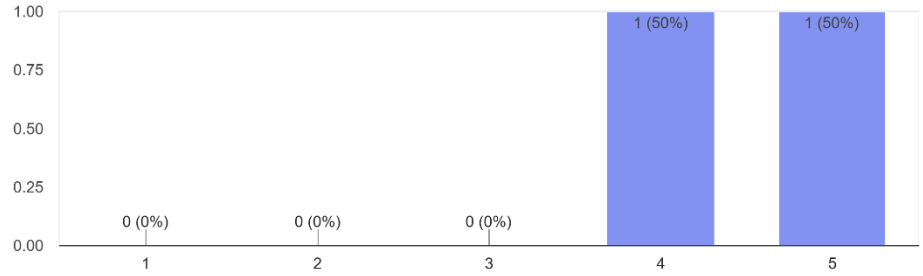
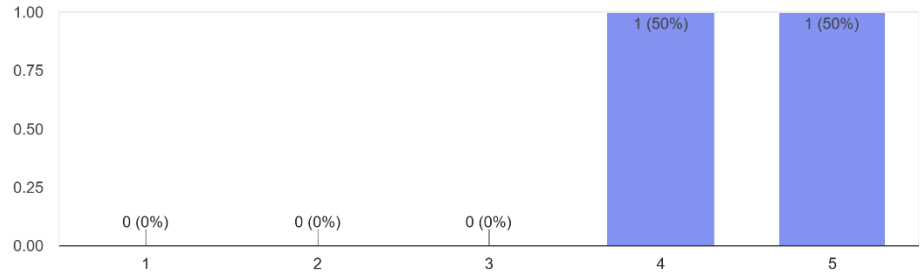
(I Wayan Suarsana)

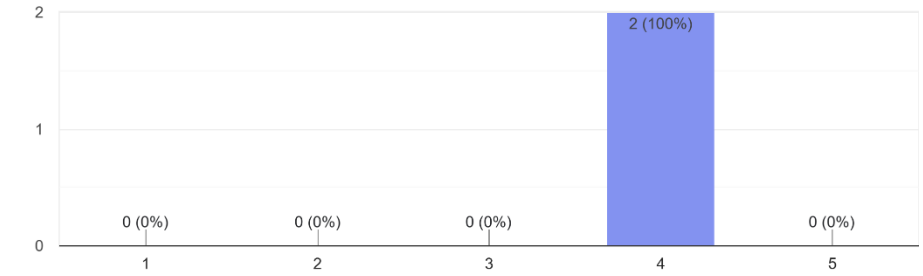
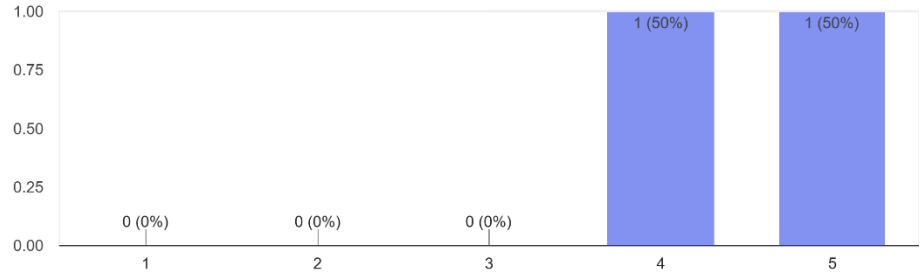
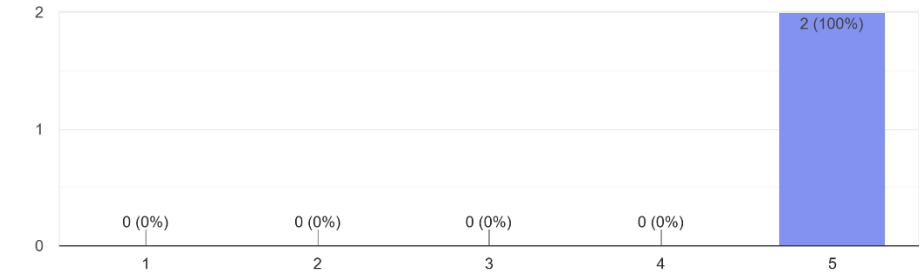
Instrumen Penguji 2

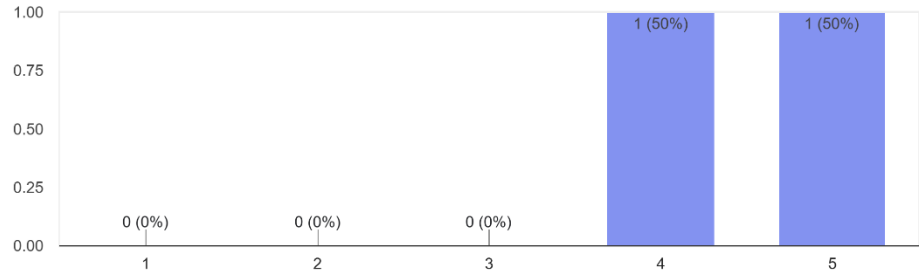
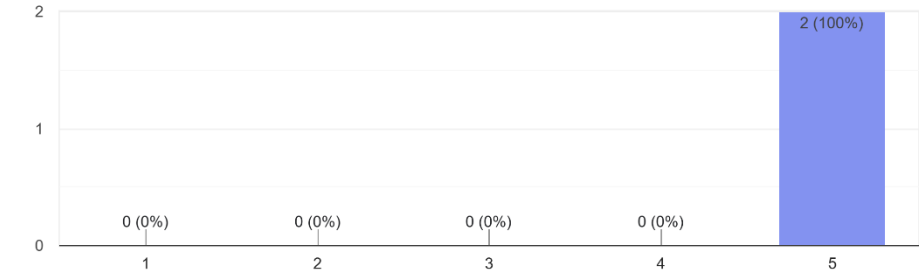
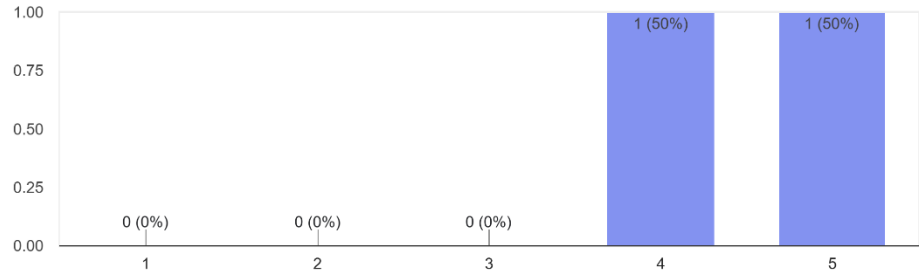
Lampiran 0.5 Hasil Uji Media

Q	Grafik Jawaban Uji Media Game
Q1	Aturan dan tujuan permainan jelas serta mudah dipahami oleh pemain. 2 responses 
Q2	Logika permainan (misal: skor, progres, kondisi menang/kalah) berjalan sesuai rencana. 2 responses 
Q3	Fungsionalitas inti (core gameplay) berjalan lancar tanpa bug. 2 responses 

Q4	Sistem kontrol (misal: tombol, sentuhan) responsif dan mudah digunakan. 2 responses 
Q5	Navigasi menu (menu utama, pengaturan, dll.) mudah dioperasikan. 2 responses 
Q6	Umpan balik (feedback) dari sistem (misal: UI dari jawaban benar) jelas dan membantu. 2 responses 

Q7	Desain karakter dan lingkungan (grafis) menarik dan konsisten secara visual. 2 responses 
Q8	Musik latar dan efek suara (audio) sesuai dengan tema dan suasana permainan. 2 responses 
Q	**Grafik Jawaban Uji Media Web**
Q1	Keberhasilan sistem mengklasifikasikan audio yang diunggah. 2 responses 

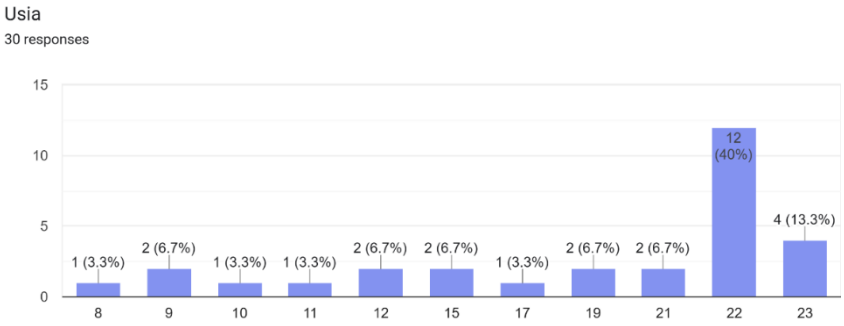
Q2	Keberhasilan sistem menghasilkan soal dalam format JSON yang valid. 2 responses 
Q3	Alur kerja end-to-end (dari unggah hingga unduh JSON) berjalan lancar tanpa eror. 2 responses 
Q4	Langkah-langkah untuk menggunakan fitur jelas dan tidak berbelit-belit. 2 responses 

Q5	Antarmuka pengguna (UI) mudah dipahami dan intuitif. 2 responses 
Q6	Informasi dan instruksi yang disajikan pada web mudah dimengerti. 2 responses 
Q7	Tata letak (layout) elemen pada halaman terstruktur dengan baik. 2 responses 

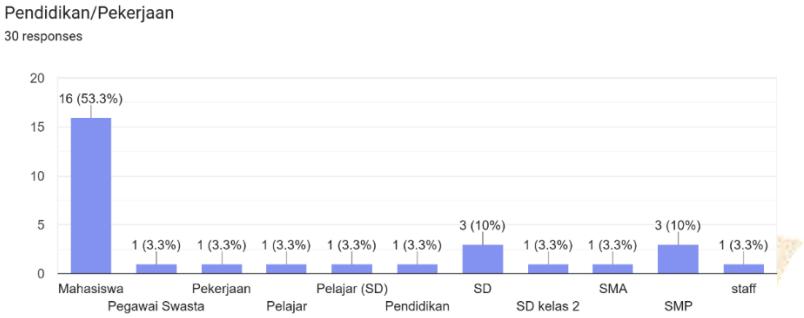
Lampiran 0.6 Hasil Uji Respon Pengguna

UEQ Tools **:** https://github.com/Risvaaa12/Harmoni-Nusantara-Aset/blob/f285352e6b1b622b06296e4730b8cc7aa91882dd/Short_UEQ_Data_Analysis_Tool.xlsx

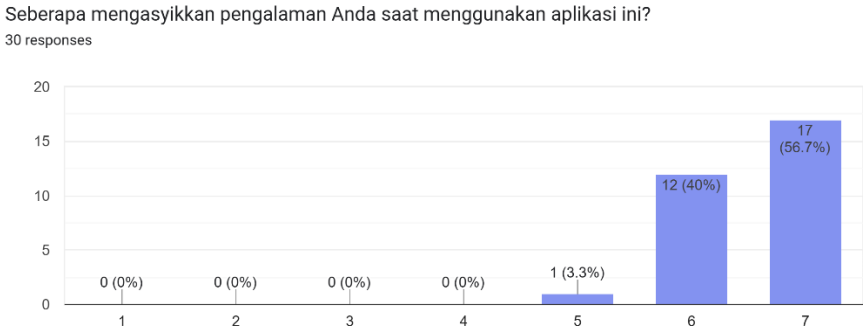
Nama	Usia	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Skor
Ni Luh Eka Ari Laksmini	19	6	6	7	7	6	6	7	6	51
I Komang Ari Adnyana	12	6	5	6	5	7	6	5	7	47
Kadek alya Fitri yani dewi	12	7	7	7	7	7	6	7	7	55
I Kadek Ary Juniada	19	7	7	7	7	7	7	7	7	56
Sakha Darma Adikara	8	6	6	7	6	6	6	6	6	49
Agus Harnata	15	7	5	5	6	7	6	5	6	47
Kadek Krishna Kindersley	10	7	7	7	7	7	7	7	7	56
Desi	9	7	7	7	7	7	7	7	7	56
Ni Komang Novi Candani	9	7	7	7	7	7	7	7	7	56
Ni Kadek Suci Andani	17	7	7	7	7	7	7	7	7	56
Ni putu Devi librayanti	11	7	7	7	5	7	7	7	2	49
Prima Nugraha	15	7	7	7	7	7	7	7	7	56
Ni Kadek Ayu Meriandini	22	7	7	7	7	7	7	7	7	56
Aditya Indra	23	7	7	7	7	7	7	7	7	56
Yandex bross	22	6	6	7	6	5	6	6	7	49
Reiki	22	6	7	6	6	6	7	7	6	51
I Nyoman Wage Wira Wardana	22	6	6	6	7	6	7	6	7	51
I Kadek Prasta Yudhantara	23	7	7	7	7	7	7	5	7	54
Putu Adi Sastrawan	22	6	7	5	5	5	6	6	6	46
Waradiana	22	7	7	7	7	7	7	7	7	56
Yuda	22	7	7	7	7	7	7	7	6	55
Agus Tamayasa	22	7	7	7	7	6	7	6	7	54
Kadek Puja Astawa	21	7	7	6	7	6	7	6	7	53
Ngakan Gde Satria Abirama	22	7	7	6	5	6	6	7	7	51
I Made Adi Wahyudinata	22	6	7	5	6	7	7	7	7	52
Made Kembar Sariasa	23	6	7	7	6	6	7	6	7	52
Putu Sava Adikara Budi	22	5	7	6	6	5	6	7	7	49
Komang Wibisana	21	6	7	6	7	7	7	6	6	52
Ni Komang Meliyanti	23	6	7	6	7	6	7	6	7	52
Kannha Wiswambara	22	6	7	7	7	6	6	7	7	53
AVG		6,53	6,7	6,53	6,5	6,47	6,7	6,5	6,6	**52,5**
SUM		196	202	196	195	194	200	195	198	**1576**

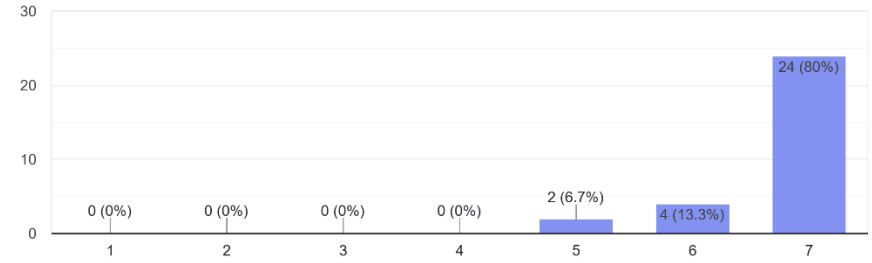
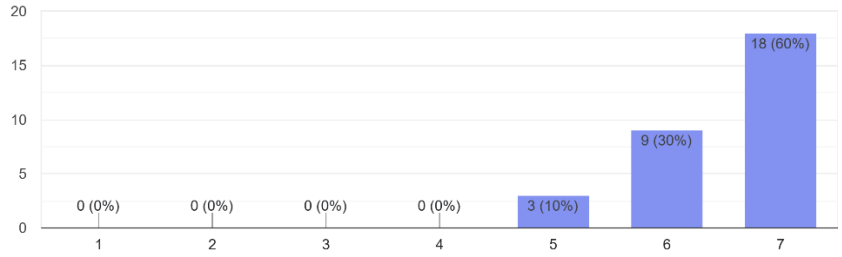
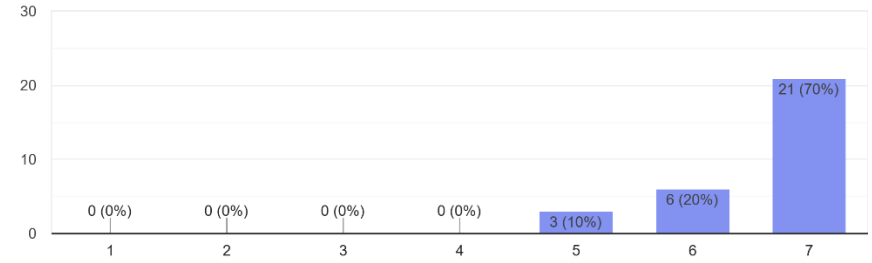


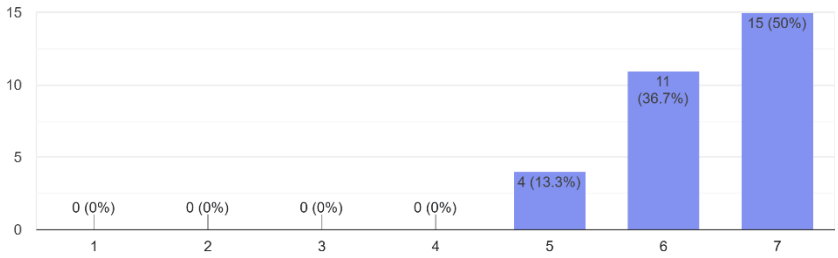
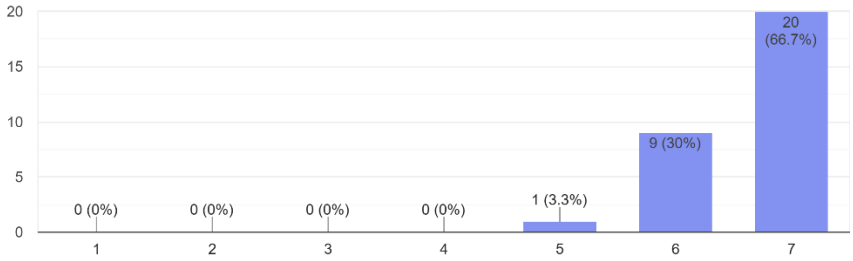
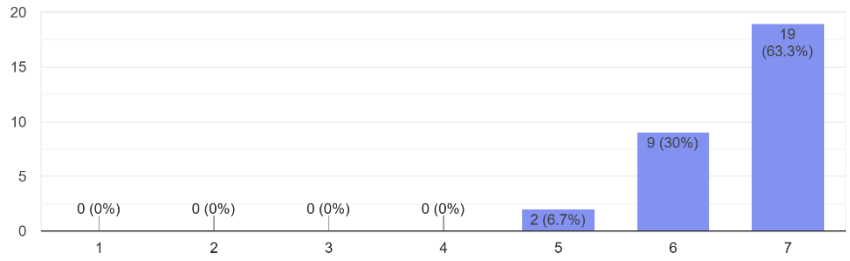
Grafik Usia



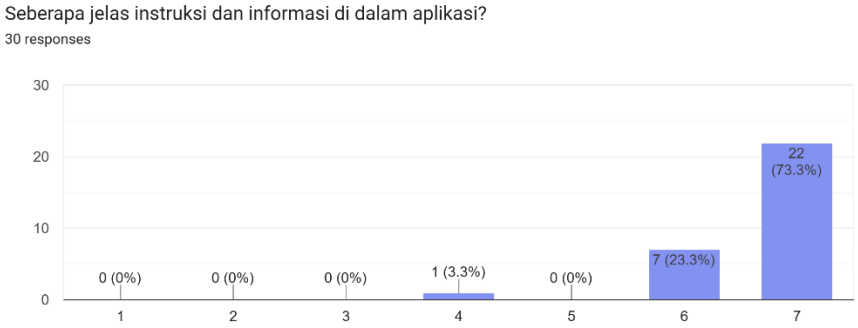
Grafik Pendidikan/Pekerjaan

Q	Grafik Jawaban Responden
Q1	

Q2	Seberapa sederhana penggunaan aplikasi ini? 30 responses 
Q3	Seberapa menarik tampilan dan desain aplikasi ini? 30 responses 
Q4	Apakah aplikasi ini terasa mendukung aktivitas Anda dalam belajar gamelan Bali? 30 responses 

Q5	Menurut Anda, seberapa kreatif aplikasi ini? 30 responses 
Q6	Bagaimana Anda menilai alur penggunaan aplikasi ini? 30 responses 
Q7	Bagaimana kesan Anda terhadap desain aplikasi ini? 30 responses 

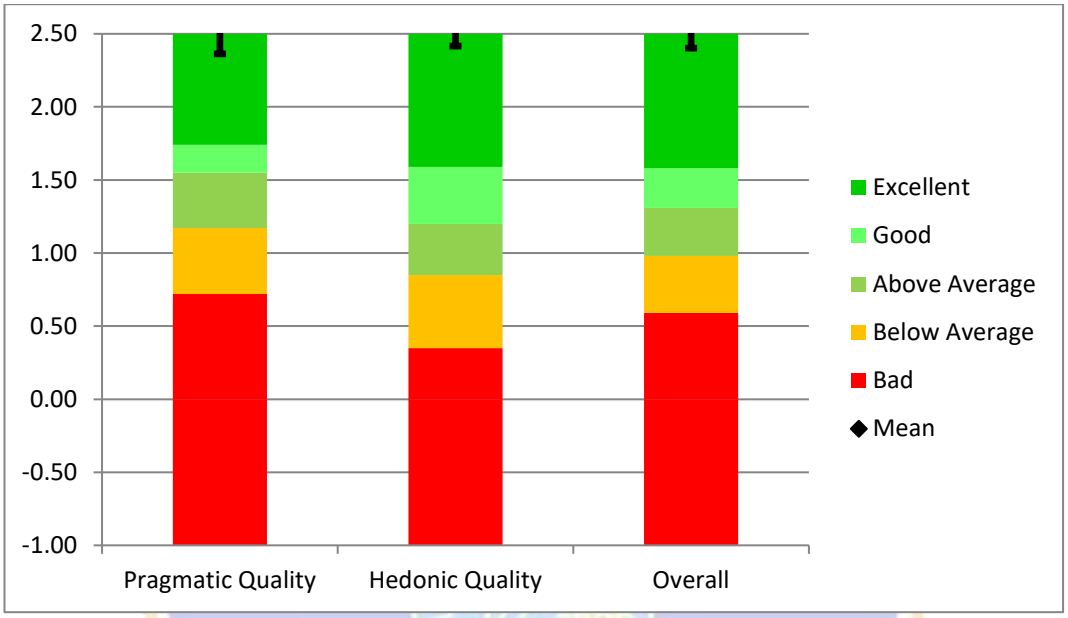
Q8	



Skale means per person		
Pragmatic Quality	***Hedonic Quality***	***Overall***
1,50	2,25	1,88
1,50	2,25	1,88
3,00	2,75	2,88
3,00	3,00	3,00
2,25	2,00	2,13
1,75	2,00	1,88
3,00	3,00	3,00
3,00	3,00	3,00
3,00	3,00	3,00
3,00	3,00	3,00
2,50	1,75	2,13
3,00	3,00	3,00
3,00	3,00	3,00
3,00	3,00	3,00
2,25	2,00	2,13
2,25	2,50	2,38
2,25	2,50	2,38
3,00	2,50	2,75
1,75	1,75	1,75
3,00	3,00	3,00
3,00	2,75	2,88
3,00	2,50	2,75
2,75	2,50	2,63
2,25	2,50	2,38
2,00	3,00	2,50
2,50	2,50	2,50
2,00	2,25	2,13

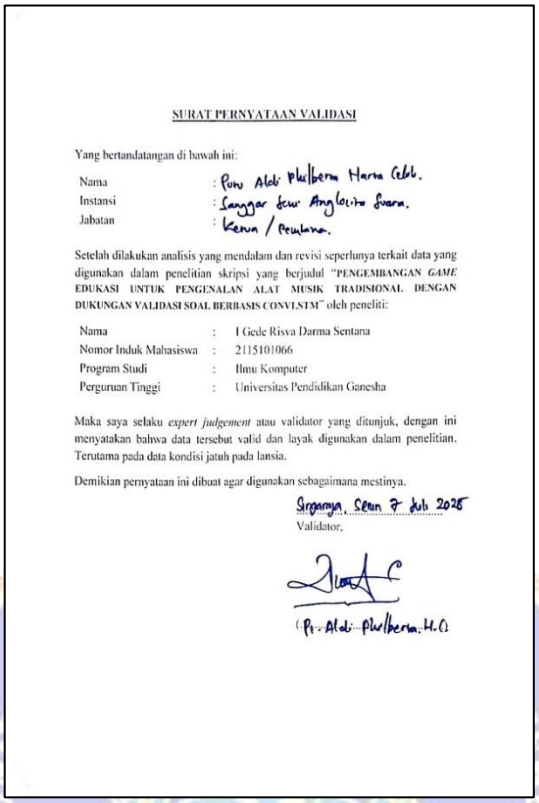
Skale means per person		
Pragmatic Quality	***Hedonic Quality***	***Overall***
2,50	2,50	2,50
2,50	2,50	2,50
2,75	2,50	2,63

Sakala Rata-rata Hasil Transformasi Data Per Responden



Grafik Hasil Komparasi Benchmark UEQ

Lampiran 0.7 Surat Pernyataan Validitas Dataset



SURAT PERNYATAAN VALIDASI

Yang bertandatangan di bawah ini:

Nama : Pon Aldi Pluiberta Harta Cebl.
Instansi : Sanggar tcw. Anglaiita Suara.
Jabatan : Kerua / Pemlatna.

Setelah dilakukan analisis yang mendalam dan revisi seperlunya terkait data yang digunakan dalam penelitian skripsi yang berjudul "**PENGEMBANGAN *GAME* EDUKASI UNTUK PENGENALAN ALAT MUSIK TRADISIONAL DENGAN DUKUNGAN VALIDASI SOAL BERBASIS CONVLSTM**" oleh peneliti:

Nama	:	I Gede Risva Darma Sentana
Nomor Induk Mahasiswa	:	2115101066
Program Studi	:	Ilmu Komputer
Perguruan Tinggi	:	Universitas Pendidikan Ganesha

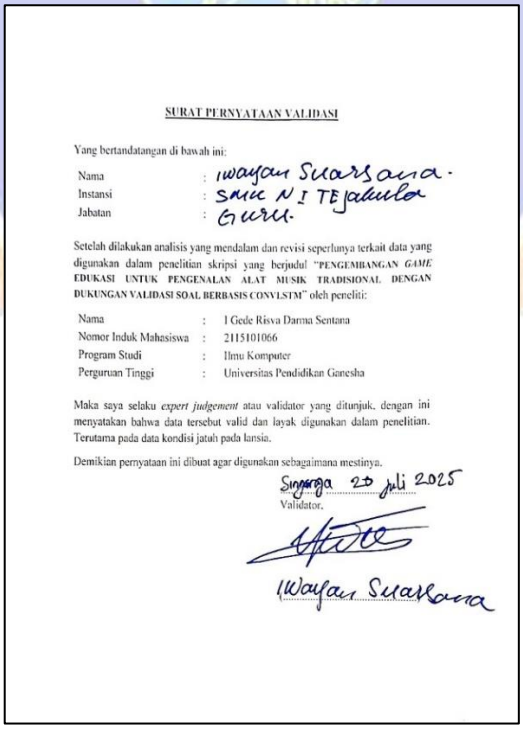
Maka saya selaku *expert judgement* atau validator yang ditunjuk, dengan ini menyatakan bahwa data tersebut valid dan layak digunakan dalam penelitian. Terutama pada data kondisi jatuh pada lansia.

Demikian pernyataan ini dibuat agar digunakan sebagaimana mestinya.

Singaraja, Senin 7 Juli 2025
Validator,

(Pt. Aldi Pluiberta. H.C)

Validator 1



SURAT PERNYATAAN VALIDASI

Yang bertandatangan di bawah ini:

Nama : Iwayan Suarsana.
Instansi : SMK N 1 TEjakula
Jabatan : Guru.

Setelah dilakukan analisis yang mendalam dan revisi seperlunya terkait data yang digunakan dalam penelitian skripsi yang berjudul "**PENGEMBANGAN *GAME* EDUKASI UNTUK PENGENALAN ALAT MUSIK TRADISIONAL DENGAN DUKUNGAN VALIDASI SOAL BERBASIS CONVLSTM**" oleh peneliti:

Nama	:	I Gede Risva Darma Sentana
Nomor Induk Mahasiswa	:	2115101066
Program Studi	:	Ilmu Komputer
Perguruan Tinggi	:	Universitas Pendidikan Ganesha

Maka saya selaku *expert judgement* atau validator yang ditunjuk, dengan ini menyatakan bahwa data tersebut valid dan layak digunakan dalam penelitian. Terutama pada data kondisi jatuh pada lansia.

Demikian pernyataan ini dibuat agar digunakan sebagaimana mestinya.

Singaraja 20 Juli 2025
Validator.

IWayan Suarsana

Validator 2

Lampiran 0.8 Susunan Soal

Distribusi Per Level:

Level	Tema & Reward	Soal Teks	Soal Gambar	Soal Audio	Total Soal
1	Gong	8 (3 umum + 3 tema + 2 alat gamelan/musik gamelan)	2 (1 tema + 1 alat gamelan lain)	0	10
2	Pemade	8 (3 umum + 3 tema + 2 alat gamelan/musik gamelan)	2 (1 tema + 1 alat gamelan lain)	0	10
3	Kemong	8 (3 umum + 3 tema + 1 alat gamelan/musik gamelan)	2 (1 tema + 1 alat gamelan lain)	0	10
4	Kantilan	8 (4 umum + 2 tema + 3 alat gamelan/musik gamelan)	2 (1 tema + 1 alat gamelan lain)	3	11
5	Kempur	6 (1 umum + 2 tema + 3 alat gamelan/musik gamelan)	2 (1 tema + 1 alat gamelan lain)	3	11
6	Jublag	5 (2 tema + 3 alat gamelan/musik gamelan)	3 (1 tema + 2 alat gamelan lain)	4	12
7	Kempli	5 (2 tema + 3 alat gamelan/musik gamelan)	3 (1 tema + 2 alat gamelan lain)	5	13
8	Penyahcah	5 (2 tema + 3 alat	4 (2 tema + 2 alat	5	14

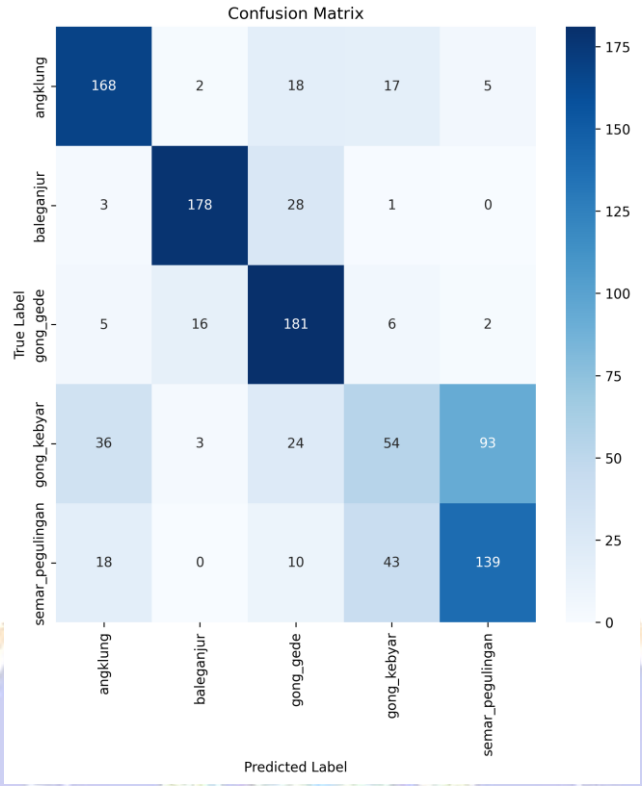
		gamelan/musik gamelan)	gamelan lain)		
Total:		53	20	20	91

Soal Audio:
Hanya menggunakan klip yang diambil dari salah satu dari lima genre:

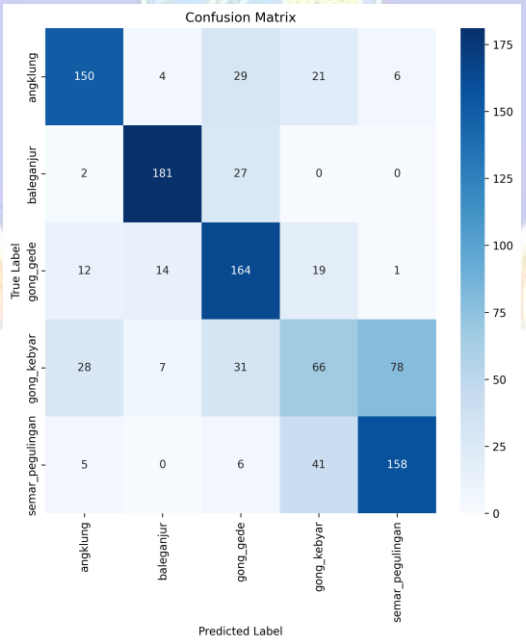
- Angklung
- Balaganjur
- Semar Pegulingan
- Gong Kebyar
- Gong Gede

Full Soal: **https://github.com/Risvaaa12/Harmoni-Nusantara-Aset.git**

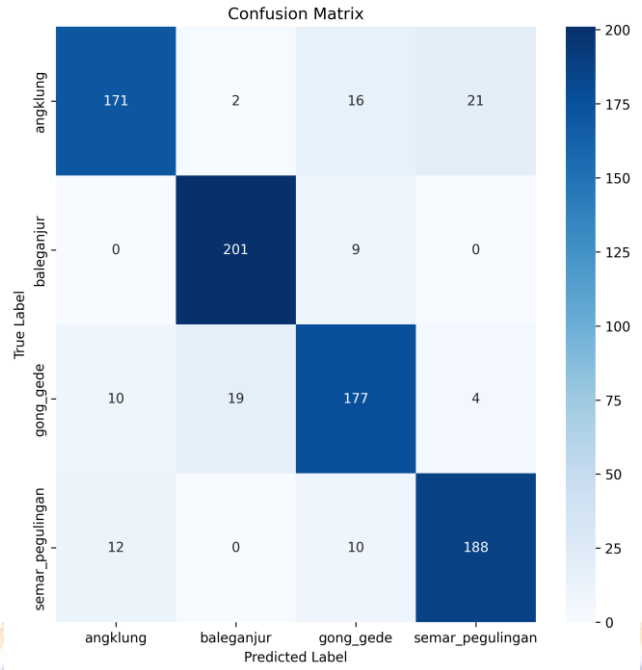
Lampiran 0.9 Dokumentasi Training (*Trial & Error)*



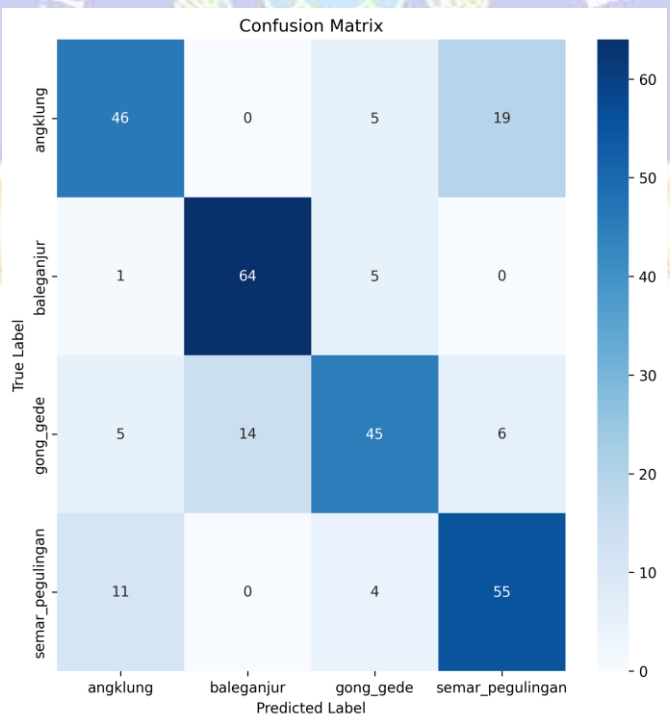
Confussion Matrix Training Augmentasi Dataset 10 Detik



Confussion Matrix Training Data Raw



Confussion Matrix Training Tanpa Gong Kebyar Dataset 10 Detik



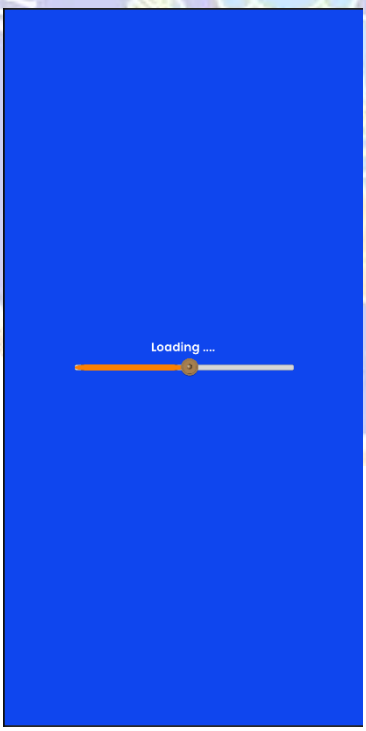
Confussion Matrix Tanpa Gong Kebyar Dataset 15 Detik

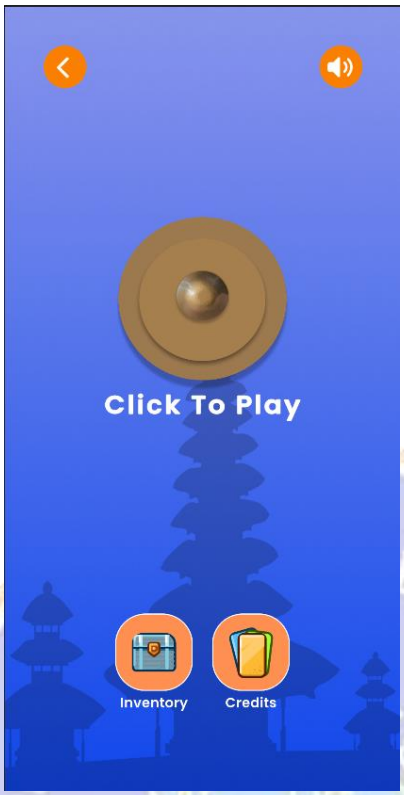
Lampiran 0.10 Dokumentasi Uji dan Validasi Dataset

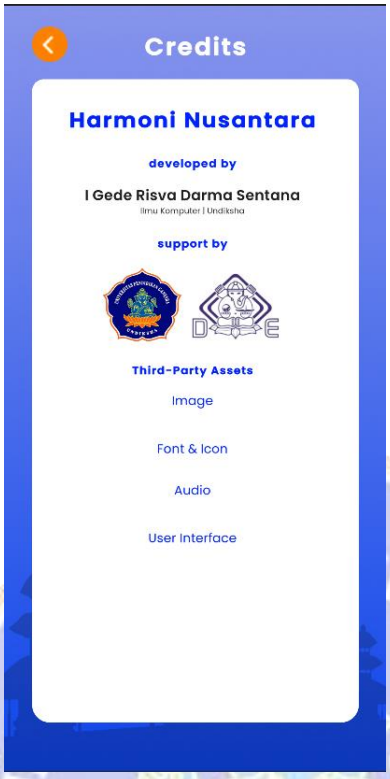
Dokumentasi Uji Ahli Isi dan Validasi Dataset

Dokumentasi Uji Respon Pengguna

Lampiran 0.11 User Interface Game

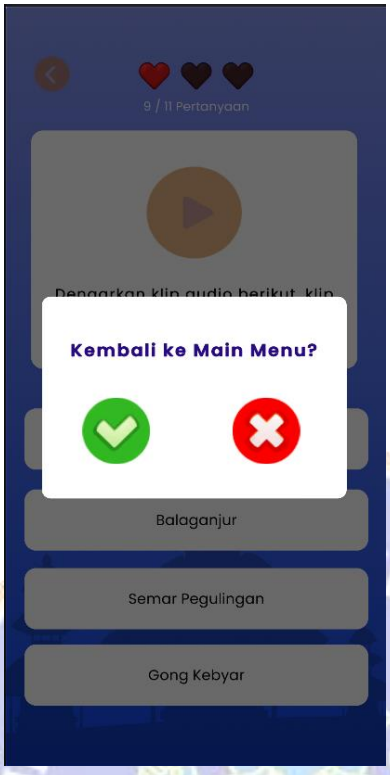
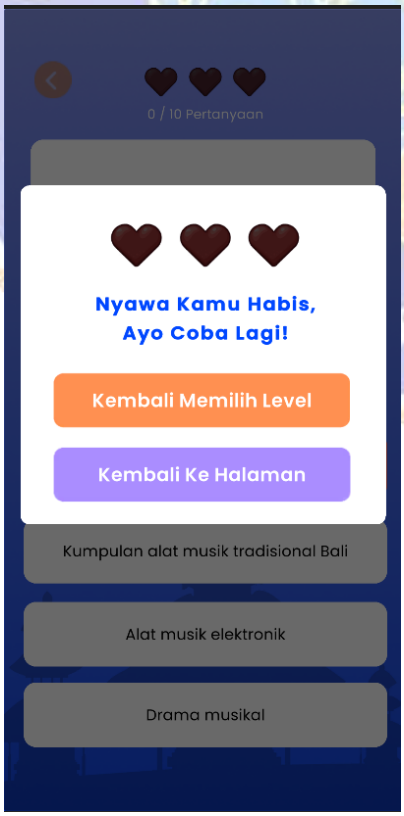
No	UI	Keterangan/Fungsi
1		*Splash Screen*: Tampilan ini muncul secara otomatis sesaat setelah pemain menjalankan aplikasi untuk pertama kalinya. Layar ini hanya bersifat sementara dan berfungsi untuk membangun identitas visual awal game dengan menampilkan logo "Harmoni Nusantara" dan logo pengembang. Tujuannya adalah memberikan kesan pertama yang profesional sebelum pemain masuk ke dalam pengalaman game yang sesungguhnya.
2		*Loading screen*: Layar ini berfungsi sebagai jembatan transisi antar-scene yang membutuhkan waktu untuk memuat aset (seperti gambar, audio, atau data level). Tujuannya adalah untuk menjaga pengalaman pengguna (UX) agar tidak terasa terputus atau terkesan "macet" saat menunggu. Layar ini biasanya dilengkapi elemen visual dinamis seperti bar kemajuan (*progress bar*), animasi ikon, atau tips singkat terkait permainan untuk membuat waktu tunggu terasa lebih singkat dan informatif.

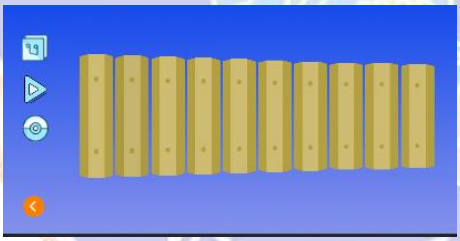
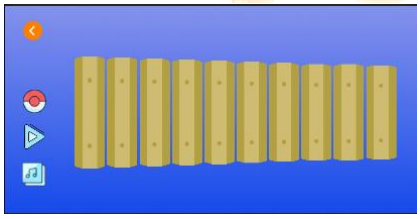
No	UI	Keterangan/Fungsi
3		*Home screen*: Sebagai pusat navigasi utama aplikasi, layar ini menyambut pemain setelah proses pemuatan awal selesai. Latar belakang musik khas Nusantara langsung diputar untuk membangun suasana. Pengguna disajikan dengan beberapa opsi interaktif utama, tombol Play untuk masuk ke peta level, tombol Inventory yang didesain menonjol untuk menarik perhatian ke fitur koleksi, tombol Credits untuk melihat informasi pengembang, serta ikon fungsional seperti Mute untuk mengontrol audio dan Keluar untuk menutup aplikasi.
4		*Inventory screen*: Halaman ini berfungsi sebagai galeri virtual yang memamerkan pencapaian pemain. Alat musik yang telah berhasil dikumpulkan sebagai *reward* ditampilkan dalam bentuk kartu atau ikon yang menarik secara visual. Setiap item di inventaris dapat dipilih oleh pemain, yang kemudian akan memunculkan informasi detail mengenai alat musik tersebut serta tombol untuk memainkannya, sehingga pemain dapat mendengarkan kembali suara khas dari instrumen yang telah ia menangkan.

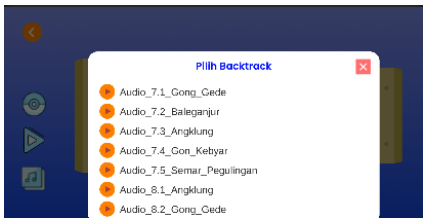
No	UI	Keterangan/Fungsi
5		*Credits screen:* Layar ini untuk memberikan atribusi dan informasi mengenai pihak-pihak yang terlibat dalam pembuatan game. Berisi daftar nama tim pengembang, ucapan terima kasih kepada pihak yang mendukung, serta lisensi atau atribusi untuk aset pihak ketiga (seperti musik, gambar, atau *font*) yang digunakan. Halaman ini menunjukkan transparansi dan profesionalisme proyek.
6		*Map screen:* Layar ini merupakan visualisasi dari progres perjalanan pemain dalam menjelajahi kekayaan musik Nusantara, dimulai dari provinsi Bali. Level-level direpresentasikan sebagai titik-titik di atas peta. Level yang sudah diselesaikan ditandai secara visual (misalnya, dengan warna cerah atau tanda bintang), sementara level yang belum terbuka ditampilkan terkunci. Terdapat navigasi Next dan Back untuk berpindah antar *stage* (kelompok level). Tombol level utama, yang direpresentasikan dengan angka, menjadi gerbang untuk memulai tantangan kuis pada level tersebut.

No	UI	Keterangan/Fungsi
7		*Narrative screen:* Halaman ini berfungsi sebagai pengantar naratif yang muncul di awal setiap level. Tampilan ini menyajikan cuplikan cerita singkat atau petunjuk tematik yang relevan dengan konten soal di level tersebut (misalnya, pengenalan singkat tentang jenis gamelan yang akan menjadi fokus). Tujuannya adalah untuk membangun konteks, memberikan pengalaman yang lebih mendalam (*immersion*), dan mempersiapkan pemain untuk tantangan yang akan datang.
8		*Quiz screen:* Halaman ini adalah layar inti dari permainan di mana interaksi edukatif utama terjadi. Desainnya berfokus pada kejelasan dan kemudahan interaksi, menampilkan area soal di bagian atas, empat opsi jawaban di bagian tengah, serta status skor dan sisa nyawa di bagian sudut layar.

No	UI	Keterangan/Fungsi
9		*Quiz Screen:* Kondisi pemain ketika jawaban salah, umpan balik visual yang diberikan bersifat korektif. Opsi yang salah akan berubah warna menjadi merah, dan jumlah nyawa pemain akan berkurang satu. Ini memberi tahu pemain secara instan bahwa pilihan mereka tidak tepat.
10		*Quiz Screen:* Kondisi pemain ketika pemain memilih jawaban yang benar, sistem memberikan umpan balik visual positif. Opsi jawaban yang dipilih akan berubah warna menjadi hijau.

No	UI	Keterangan/Fungsi
11		*Pop Up Confirm Panel:* Panel ini adalah elemen UI modal yang muncul di atas layar utama ketika pemain menekan tombol yang berpotensi menghentikan progres, seperti tombol 'kembali' di tengah permainan. Panel ini meminta konfirmasi (misalnya, "Apakah Anda yakin ingin keluar dari permainan?") untuk mencegah pemain keluar secara tidak sengaja dan kehilangan kemajuan mereka.
12		*Pop Up Failed Panel:* Panel modal ini muncul secara otomatis ketika pemain kehabisan semua nyawa dalam satu level. Tampilan ini secara jelas menginformasikan bahwa pemain telah gagal, dan memberikan dua pilihan tindakan yang jelas, tombol untuk Mengulang Level atau tombol untuk Kembali ke Halaman Peta.

No	UI	Keterangan/Fungsi
13		*Pop Up Reward Panel:* Sebagai bentuk perayaan, panel ini muncul ketika pemain berhasil menyelesaikan semua tantangan dalam satu level. Panel ini menampilkan visual dari alat musik yang berhasil didapatkan sebagai *reward*. Pemain diberikan tiga pilihan, tombol untuk langsung Memainkan suara alat musik tersebut, tombol untuk Lanjut ke level berikutnya, atau tombol untuk Kembali ke Halaman Utama.
12		Alat Musik *screen:* Halaman ini berfungsi untuk menampilkan hadiah yang dipilih atau didapatkan dan dapat dimainkan oleh pemain. Selain itu juga ada tombol kembali ke halaman utama. Halaman ini menyesuaikan kondisi yang dilakukan pemain.
13		Alat Musik *Screen:* Kondisi ketika pemain menekan tombol *recording,* maka tombol tersebut berubah menjadi merah dan mulai merekam suara permainan yang nantinya tersimpan.
14		Pop Up Daftar rekaman*:* Kondisi ketika pemain menekan tombol segitiga untuk membuka daftar rekaman untuk melihat dan

No	UI	Keterangan/Fungsi
		mendengarkan hasil rekaman.
15		Pop Up Daftar *Backtrack* Gamelan: Kondisi ketika pemain menekan tombol kotak (*playlist*) untuk memainkan *backtrack* gamelan.

Lampiran 0.12 Soal JSON

Full Soal: https://github.com/Risvaaa12/Harmoni-Nusantara-Aset/blob/c320eda6e1ef35b33ba66d2722ce14efc24f748d/susunan_soal.json


```
"level": 1,
            "questions_in_level": [
              {
                "id": 1,
                "path_audio": null,
                "path_gambar": null,
                "teks_pertanyaan": "Gamelan Bali merupakan...",
                "opsi_jawaban": {
                  "a": "Kumpulan tarian modern",
                  "b": "Kumpulan alat musik tradisional Bali",
                  "c": "Alat musik elektronik",
                  "d": "Drama musikal"
                },
                "kunci_jawaban_dokumen": "b"
              },
              {
                "id": 2,
                "path_audio": null,
                "path_gambar": null,
                "teks_pertanyaan": "Apa fungsi utama gamelan Bali dalam budaya Bali?",
                "opsi_jawaban": {
                  "a": "Untuk hiburan",
                  "b": "Untuk pengiring upacara dan tarian adat",
                  "c": "Untuk dibanggakan",
                  "d": "Untuk kompetisi"
                },
                "kunci_jawaban_dokumen": "b"
              },
              {
                "id": 3,
                "path_audio": null,
                "path_gambar": null,
                "teks_pertanyaan": "Apa nama satu set lengkap dari gamelan Bali?",
                "opsi_jawaban": {
                  "a": "Band tradisional",
                  "b": "Orkestra",
                  "c": "Bebarungan",
                  "d": "Paduan Gamelan"
                },
                "kunci_jawaban_dokumen": "c"
              },
              {
                "id": 4,
                "path_audio": null,
                "path_gambar": "/aset/1.1_Kendang.webp",
                "teks_pertanyaan": "Dalam bebarungan satu perangkat sebagai perkusi,
perangkat pada gambar dimainkan dengan cara...",
                "opsi_jawaban": {
                  "a": "Ditiup",
                  "b": "Dipetik",
                  "c": "Dipukul",
                  "d": "Digesek"
```

```
      },
      "kunci_jawaban_dokumen": "c"
    },
    {
      "id": 5,
      "path_audio": null,
      "path_gambar": "/aset/None",
      "teks_pertanyaan": "Apa nama alat musik pada gambar sebelumnya dan
berperan untuk apakah alat musik tersebut…",
      "opsi_jawaban": {
        "a": "Kendang - Menentukan ritme dan tempo",
        "b": "Kendang – Sebagai melodi",
        "c": "Gong - Menentukan ritme dan tempo",
        "d": "Cengceng Kecek – Penyatu irama"
      },
      "kunci_jawaban_dokumen": "a"
    },
    {
      "id": 6,
      "path_audio": null,
      "path_gambar": "/aset/None",
      "teks_pertanyaan": "Selain kendang, Cengceng Kecek hadir sebagai penyatu
irama. Cengceng Kecek menghasilkan suara yang…",
      "opsi_jawaban": {
        "a": "Menambah efek ritmis",
        "b": "Menghasilkan melodi utama",
        "c": "Hanya sebagai hiasan",
        "d": "Menambah tempo"
      },
      "kunci_jawaban_dokumen": "a"
    },
    {
      "id": 7,
      "path_audio": null,
      "path_gambar": "/aset/1.2_Gong.webp",
      "teks_pertanyaan": "Alat apakah ini? Apa fungsinya?",
      "opsi_jawaban": {
        "a": "Suling - Melodi",
        "b": "Gong - Penanda siklus",
        "c": "Kempur - Penanda siklus",
        "d": "Cengceng Kecek - Penyatu irama"
      },
      "kunci_jawaban_dokumen": "b"
    },
    {
      "id": 8,
      "path_audio": null,
      "path_gambar": "/aset/None",
      "teks_pertanyaan": "Dari gambar sebelumnya, ciri-ciri gong yang benar dibawah
ini, kecuali...",
      "opsi_jawaban": {
        "a": "Terbuat dari perunggu",
        "b": "Berwarna putih",
        "c": "Berbentuk bundar",
        "d": "Terdapat pencon atau moncol di tengah-tengah"
      },
      "kunci_jawaban_dokumen": "b"
    },
```

```
            {
              "id": 9,
              "path_audio": null,
              "path_gambar": "/aset/None",
              "teks_pertanyaan": "Berdasrkan gambar sebelumnya, Gong dimainkan dengan
cara?",
              "opsi_jawaban": {
                "a": "Ditiup",
                "b": "Dipetik",
                "c": "Dipukul",
                "d": "Digesek"
              },
              "kunci_jawaban_dokumen": "c"
            },
            {
              "id": 10,
              "path_audio": null,
              "path_gambar": null,
              "teks_pertanyaan": "Suara khas Gong Ageng terdengar dalam bentuk resonansi
yang?",
              "opsi_jawaban": {
                "a": "Ringan dan cepat",
                "b": "Dalam, berat, dan resonan",
                "c": "Sangat tinggi",
                "d": "Mirip dengan kendang"
              },
              "kunci_jawaban_dokumen": "b"
            }
          ]
        },
          ]
        }
```

Lampiran 0.13 Kode ConvLSTM dan Web

DEVELOP MODEL: https://github.com/Risvaaa12/ConvLSTM-Development-Model.git (File: ConvLSTM)

```
# 📻 Klasifikasi Genre Gamelan dengan ConvLSTM


#1. Instalasi Library

import os
import numpy as np
import librosa
import librosa.display
import matplotlib.pyplot as plt
import seaborn as sns
import random
import pandas as pd
import noisereduce as nr
import joblib
import json
import IPython.display as ipd
from datetime import datetime
import tensorflow as tf
from scipy import signal
from tqdm import tqdm
from sklearn.preprocessing import LabelEncoder
from sklearn.manifold import TSNE
from tensorflow.keras import layers, Model, regularizers
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint,
ReduceLROnPlateau, CSVLogger
## 3. Set Paths & Hyperparameters

#Memastikan konsistensi data
seed_value = 42

os.environ['PYTHONHASHSEED'] = str(seed_value)
random.seed(seed_value)
np.random.seed(seed_value)
tf.random.set_seed(seed_value)

# Paths
train_dir = "E:/RISVA/audio_model/dataset/segmentation_10s/train/"
val_dir = "E:/RISVA/audio_model/dataset/segmentation_10s/validation/"
save_model_dir = "E:/RISVA/audio_model/model_logs/V13_fix_ast"
os.makedirs(save_model_dir, exist_ok=True)
```

```
# Audio parameters
SR = 22050
DURATION = 10
N_FFT = 1024
HOP_LENGTH = 512

# Model parameters
N_MFCC = 40
MAX_PAD_LEN = 432
BATCH_SIZE = 32
EPOCHS = 500
LEARNING_RATE = 1e-4
L2_REG = 1e-4

# # Denoising parameters
# NOISE_REDUCTION_PARAMS = {
#     'stationary': True,
#     'prop_decrease': 0.9,
#     'n_fft': N_FFT,
#     'win_length': HOP_LENGTH
# }


#4. Denoising, Augmentasi, & Ekstraksi Fitur
#Denoising
def denoise_audio(y, sr):
    # 1. Deteksi noise dari bagian non-aktif
    energy = librosa.feature.rms(y=y)
    frames = np.where(energy < np.percentile(energy, 10))[0]
    noise_samples = y[frames]


    # --- Versi Perbaikan ---
    GAMELAN_NR_PARAMS = {
        'stationary': False,
        'prop_decrease': 0.30,
        'n_fft': N_FFT,
        'win_length': N_FFT,
        'hop_length': N_FFT // 4,
        'use_tqdm': False
    }
    return nr.reduce_noise(
        y=y,
        y_noise=noise_samples,
        sr=sr,
        **GAMELAN_NR_PARAMS
    )
```

```
def gamelan_style_augmentation(y, sr):
    prob = 0.5  # Probabilitas tiap augmentasi

    if random.random() < prob:
        y = tempo_preserve_pitch(y, sr)

    if random.random() < prob:
        y = gamelan_pitch_shift(y, sr)

    if random.random() < prob:
        y = dynamic_range_augmentation(y)

    if random.random() < prob:
        y = cyclic_rhythm_crop(y, sr)

    return y


def tempo_preserve_pitch(y, sr):
    rate = np.random.choice([0.9, 1.0, 1.1])
    y = librosa.effects.time_stretch(y, rate=rate)
    return y


def gamelan_pitch_shift(y, sr):
    steps = np.random.choice([-0.25, 0, 0.25])
    y = librosa.effects.pitch_shift(y, sr=sr, n_steps=steps)
    return y


def dynamic_range_augmentation(y, sr=None):
    if np.random.rand() > 0.5:
        y = signal.lfilter([1, 0.5], [1], y)
    else:
        y = librosa.util.normalize(y) * 0.8
    return y


def cyclic_rhythm_crop(y, sr):
    cycle_duration = np.random.uniform(6, 8)
    samples = int(cycle_duration * sr)
    if len(y) > samples:
        start = np.random.randint(0, len(y) - samples)
        y = y[start:start+samples]
    return y
def extract_features(y, sr, augment=True):
    # Denoising
    y_clean = denoise_audio(y, sr)
```

```
    try:
        # Augmentasi
        if augment:
            y = gamelan_style_augmentation(y_clean, sr)
        else:
            y = y_clean

        # Ekstraksi fitur
        mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=40,n_fft=2048,
hop_length=512)
        spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr)
        chroma = librosa.feature.chroma_cqt(y=y, sr=sr)

        # Gabungkan fitur
        features = np.vstack([mfcc, spectral_contrast, chroma])

        # Normalisasi per fitur
        features = (features - np.mean(features, axis=1, keepdims=True)) / \
                (np.std(features, axis=1, keepdims=True) + 1e-6)

        # Padding
        if features.shape[1] < MAX_PAD_LEN:
            n_repeats = int(np.ceil(MAX_PAD_LEN / features.shape[1]))
            features = np.tile(features, n_repeats)[:, :MAX_PAD_LEN]
        else:
            features = features[:, :MAX_PAD_LEN]

        return features.T.reshape(MAX_PAD_LEN, features.shape[0], 1, 1)

    except Exception as e:
        print(f"Error in feature extraction: {str(e)}")
        return None
# 5. Load Dataset & Visualisasi
def load_dataset(directory, augment=True):
    X = []
    y = []
    classes = sorted(os.listdir(directory))

    for cls in classes:
        cls_path = os.path.join(directory, cls)
        if not os.path.isdir(cls_path):
            continue

        files = [f for f in os.listdir(cls_path) if f.lower().endswith(('.wav', '.mp3'))]

        for fn in tqdm(files, desc=f"Loading {cls}"):
            try:
                # Load audio
```

```
            y_audio, sr = librosa.load(os.path.join(cls_path, fn), sr=SR)

            # Denoising
            y_clean = nr.reduce_noise(
                y=y_audio,
                sr=SR,
                y_noise=y_audio[:int(0.3*SR)] if len(y_audio) > int(0.3*SR) else
y_audio,
                prop_decrease=0.85
            )

            # Ekstraksi fitur
            features = extract_features(y_clean, sr, augment=augment)

            # Append ke list
            X.append(features)
            y.append(cls)

        except Exception as e:
            print(f"Error processing {fn}: {str(e)}")
            continue

    return np.array(X), np.array(y)  # Konversi ke array di akhir
def visualize_processing_pipeline():
    sample_path = os.path.join(train_dir, os.listdir(train_dir)[0],
                               os.listdir(os.path.join(train_dir, os.listdir(train_dir)[0]))[0])
    y_raw, _ = librosa.load(sample_path, sr=SR)
    y_clean = denoise_audio(y_raw, SR)

    plt.figure(figsize=(15, 6))

    # Waveform
    plt.subplot(2, 2, 1)
    librosa.display.waveshow(y_raw, sr=SR)
    plt.title('Raw Audio Waveform')

    plt.subplot(2, 2, 2)
    librosa.display.waveshow(y_clean, sr=SR)
    plt.title('Denoised Waveform')

    # MFCC Comparison
    plt.subplot(2, 2, 3)
    mfcc_raw = librosa.feature.mfcc(y=y_raw, sr=SR, n_mfcc=40)
    librosa.display.specshow(mfcc_raw, x_axis='time')
    plt.title('Raw MFCC')

    plt.subplot(2, 2, 4)
    mfcc_clean = librosa.feature.mfcc(y=y_clean, sr=SR, n_mfcc=40)
```

```
    librosa.display.specshow(mfcc_clean, x_axis='time')
    plt.title('Denoised MFCC')
    plt.tight_layout()

    nama_file_cm = 'denoising.png'
    path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)

    try:
        plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
        print(f"Berhasil disimpan")
    except Exception as e:
        print(f"Gagal menyimpan")


    plt.show()

# Jalankan visualisasi pipeline
print("Visualisasi Proses Preprocessing:")
visualize_processing_pipeline()
def visualize_gamelan_augmentations():
    # Ambil sample audio
    sample_cls = random.choice(os.listdir(train_dir))
    sample_path = os.path.join(train_dir, sample_cls,
random.choice(os.listdir(os.path.join(train_dir, sample_cls))))
    y, sr = librosa.load(sample_path, sr=SR)

    plt.figure(figsize=(20, 15))

    plt.subplot(5, 4, 1)
    librosa.display.waveshow(y, sr=sr)
    plt.title(f'Original Waveform\n({sample_cls})', fontsize=10)

    plt.subplot(5, 4, 2)
    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=40)
    librosa.display.specshow(mfcc, x_axis='time', cmap='coolwarm')
    plt.title('Original MFCC', fontsize=10)

    plt.subplot(5, 4, 3)
    spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr)
    librosa.display.specshow(spectral_contrast, x_axis='time', cmap='viridis')
    plt.title('Original Spectral Contrast', fontsize=10)

    plt.subplot(5, 4, 4)
    chroma = librosa.feature.chroma_cqt(y=y, sr=sr)
    librosa.display.specshow(chroma, x_axis='time', cmap='plasma')
    plt.title('Original Chroma', fontsize=10)

    # Augmentasi dan visualisasi
```

```
    augmentations = [
        ('Tempo Change', tempo_preserve_pitch),
        ('Pitch Shift', gamelan_pitch_shift),
        ('Dynamic Range', dynamic_range_augmentation),
        ('Rhythm Crop', cyclic_rhythm_crop)
    ]

    for row, (title, aug_func) in enumerate(augmentations, 2):
        y_aug = aug_func(y.copy(), sr)

        # Waveform
        plt.subplot(5, 4, row*4-3)
        librosa.display.waveshow(y_aug, sr=sr)
        plt.title(f'{title} Waveform', fontsize=10)

        # MFCC
        plt.subplot(5, 4, row*4-2)
        mfcc_aug = librosa.feature.mfcc(y=y_aug, sr=sr, n_mfcc=40)
        librosa.display.specshow(mfcc_aug, x_axis='time', cmap='coolwarm')
        plt.title(f'{title} MFCC', fontsize=10)

        # Spectral Contrast
        plt.subplot(5, 4, row*4-1)
        spectral_contrast_aug = librosa.feature.spectral_contrast(y=y_aug, sr=sr)
        librosa.display.specshow(spectral_contrast_aug, x_axis='time',
cmap='viridis')
        plt.title(f'{title} Spectral Contrast', fontsize=10)

        # Chroma
        plt.subplot(5, 4, row*4)
        chroma_aug = librosa.feature.chroma_cqt(y=y_aug, sr=sr)
        librosa.display.specshow(chroma_aug, x_axis='time', cmap='plasma')
        plt.title(f'{title} Chroma', fontsize=10)

    plt.tight_layout()
    nama_file_cm = 'augmentasi.png'
    path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)

    try:
        plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
        print(f"Berhasil disimpan")
    except Exception as e:
        print(f"Gagal menyimpan")

    plt.show()

print("Visualisasi Teknik Augmentasi Gamelan:")
visualize_gamelan_augmentations()
```

```
#visualisai distirbusi kelas
def plot_class_distribution_on_ax(ax, directory, title):
    class_counts = {}
    if not os.path.isdir(directory):
        ax.text(0.5, 0.5, f"Direktori tidak ditemukan:\n{directory}",
                horizontalalignment='center', verticalalignment='center',
                transform=ax.transAxes, color='red')
        ax.set_title(f'Error - {title}', fontsize=14)
        return

    classes = sorted(os.listdir(directory))
    if not classes:
        ax.text(0.5, 0.5, f"Tidak ada kelas ditemukan di:\n{directory}",
                horizontalalignment='center', verticalalignment='center',
                transform=ax.transAxes, color='orange')
        ax.set_title(f'Data Kosong - {title}', fontsize=14)
        return

    for cls_name in classes:
        cls_path = os.path.join(directory, cls_name)
        if os.path.isdir(cls_path):
            try:
                num_files = len([name for name in os.listdir(cls_path) if
os.path.isfile(os.path.join(cls_path, name))])
                class_counts[cls_name] = num_files
            except Exception as e:
                print(f"Tidak bisa mengakses direktori {cls_path}: {e}")
                class_counts[cls_name] = 0
        else:
            pass

    if not class_counts:
        ax.text(0.5, 0.5, f"Tidak ada data sampel ditemukan di kelas-kelas
dalam:\n{directory}",
                horizontalalignment='center', verticalalignment='center',
                transform=ax.transAxes, color='orange')
        ax.set_title(f'Data Sampel Kosong - {title}', fontsize=14)
        return

    ax.bar(class_counts.keys(), class_counts.values(), color='darkcyan')
    ax.set_title(f'Class Distribution - {title}', fontsize=14)
    ax.set_xlabel('Gamelan Genre', fontsize=12)
    ax.set_ylabel('Number of Samples', fontsize=12)
    ax.tick_params(axis='x', rotation=45, labelsize=10)
    ax.tick_params(axis='y', labelsize=10)
    ax.grid(axis='y', linestyle='--')
```

```
print("Membuat plot distribusi data Training dan Validasi...")

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(20, 8))

plot_class_distribution_on_ax(ax1, train_dir, "Training Data")
plot_class_distribution_on_ax(ax2, val_dir, "Validation Data")


fig.suptitle('Class Distribution for Training and Validation Sets', fontsize=18,
y=1.02)

plt.tight_layout(rect=[0, 0, 1, 0.98])

nama_file_dist_gabungan = 'class_distributions.png'
path_simpan_dist_gabungan = os.path.join(save_model_dir,
nama_file_dist_gabungan)

try:
    plt.savefig(path_simpan_dist_gabungan, dpi=300, bbox_inches='tight')
    print(f"Berhasil disimpan di {path_simpan_dist_gabungan}")
except Exception as e:
    print(f"Gagal menyimpan plot distribusi: {e}")

plt.show()

plt.clf()
plt.close(fig)
def plot_audio_features(file_path, class_name):
    y, sr = librosa.load(file_path, sr=SR)

    # Ekstraksi fitur
    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=N_MFCC, n_fft=N_FFT,
hop_length=HOP_LENGTH)
    spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr,
n_fft=N_FFT, hop_length=HOP_LENGTH)
    chroma = librosa.feature.chroma_cqt(y=y, sr=sr,
hop_length=HOP_LENGTH)

    # Plotting
    plt.figure(figsize=(20, 15))

    plt.subplot(5, 4, 1)
    librosa.display.waveshow(y, sr=sr)
    plt.title(f'Waveform - {class_name}')

    plt.subplot(5, 4, 2)
    librosa.display.specshow(mfcc, x_axis='time', cmap='coolwarm', sr=sr,
hop_length=HOP_LENGTH)
```

```
    plt.colorbar()
    plt.title(f'MFCC ({N_MFCC} Coefficients)')

    plt.subplot(5, 4, 3)
    librosa.display.specshow(spectral_contrast, x_axis='time', sr=sr,
cmap='cubehelix', hop_length=HOP_LENGTH)
    plt.colorbar()
    plt.title('Spectral Contrast')

    plt.subplot(5, 4, 4)
    librosa.display.specshow(chroma, x_axis='time', sr=sr, cmap='plasma',
hop_length=HOP_LENGTH)
    plt.colorbar()
    plt.title('Chroma CQT')

    plt.suptitle(f'Fitur Ekstraksi Audio - {class_name}', fontsize=16)
    plt.tight_layout()
    nama_file_cm = 'Ekstraksi fitur.png'
    path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)

    try:
        plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
        print(f"Berhasil disimpan")
    except Exception as e:
        print(f"Gagal menyimpan")

    plt.show()

for cls in sorted(os.listdir(train_dir)):
    class_path = os.path.join(train_dir, cls)
    if os.path.isdir(class_path):
        sample_files = [f for f in os.listdir(class_path) if f.endswith(('.wav',
'.mp3'))]
        if sample_files:
            plot_audio_features(os.path.join(class_path, sample_files[0]), cls)

def extract_analysis_features(file_path):
    y, sr = librosa.load(file_path, sr=SR)

    # Ekstraksi fitur dasar
    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=N_MFCC,
                                n_fft=N_FFT, hop_length=HOP_LENGTH)
    zcr = librosa.feature.zero_crossing_rate(y)
    spectral_centroid = librosa.feature.spectral_centroid(y=y, sr=sr,
                                                          n_fft=N_FFT,
                                                          hop_length=HOP_LENGTH)
    spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr,
                                                          n_fft=N_FFT,
```

```
                                        hop_length=HOP_LENGTH)
    chroma = librosa.feature.chroma_cqt(y=y, sr=sr,
                                        hop_length=HOP_LENGTH)

    # Statistik per fitur
    features = {
        'mfcc_mean': np.mean(mfcc),
        'mfcc_std': np.std(mfcc),
        'zcr_mean': np.mean(zcr),
        'spectral_centroid_mean': np.mean(spectral_centroid),
        'spectral_contrast_mean': np.mean(spectral_contrast),
        'spectral_contrast_std': np.std(spectral_contrast),
        'chroma_mean': np.mean(chroma),
        'chroma_std': np.std(chroma)
    }
    return features

# Load data dan ekstraksi
classes = ['angklung', 'baleganjur', 'gong_gede', 'gong_kebyar',
'joged_bumbung']
analysis_data = []
for cls in classes:
    cls_path = os.path.join(train_dir, cls)
    if os.path.exists(cls_path):
        for fn in tqdm(os.listdir(cls_path)[:100], desc=f"Processing {cls}"):
            if fn.lower().endswith(('.mp3', '.wav')):
                feats = extract_analysis_features(os.path.join(cls_path, fn))
                feats['class'] = cls
                analysis_data.append(feats)
    else:
        print(f"Warning: Directory not found: {cls_path}")

df = pd.DataFrame(analysis_data)

# Plot distribusi statistik fitur audio per kelas
plt.figure(figsize=(18, 12))
features_to_plot = [
    'mfcc_mean', 'zcr_mean',
    'spectral_centroid_mean', 'spectral_contrast_mean', 'chroma_mean'
]
for i, feat in enumerate(features_to_plot):
    plt.subplot(3, 2, i + 1)
    sns.violinplot(x='class', y=feat, data=df, palette='Set2')
    plt.title(feat.replace('_', ' ').title(), fontsize=12)
    plt.xticks(rotation=45)

plt.suptitle("Distribusi Statistik Fitur Audio per Kelas", fontsize=16)
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
```

```
nama_file_cm = 'distribusi statistik audio.png'
path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)

try:
    plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
    print(f"Berhasil disimpan")
except Exception as e:
    print(f"Gagal menyimpan")

plt.show()
## 6. Persiapan Dataset



# Load data
print("Memuat data training...")
X_train, y_train = load_dataset(train_dir, augment=True)
print("\nMemuat data validasi...")
X_val, y_val = load_dataset(val_dir, augment=False)

# Encode labels
encoder = LabelEncoder()
encoder.fit(y_train)
y_train_enc = encoder.transform(y_train)
y_val_enc = encoder.transform(y_val)

# Convert to one-hot
y_train_cat = tf.keras.utils.to_categorical(y_train_enc, len(encoder.classes_))
y_val_cat = tf.keras.utils.to_categorical(y_val_enc, len(encoder.classes_))
## 7. Bangun Arsitektur ConvLSTM
def build_model(input_shape, num_classes):
    inputs = tf.keras.Input(shape=input_shape)

    # Block 1
    x = layers.ConvLSTM2D(
        64,
        kernel_size=(3, 3),
        activation='tanh',
        padding='same',
        return_sequences=False,
        kernel_regularizer=regularizers.l2(L2_REG)
    )(inputs)

    # Block 2
    x = layers.Conv2D(
        128,
        kernel_size=(3, 1),
        activation='relu',
```

```
        padding='valid',
        kernel_regularizer=regularizers.l2(L2_REG)
    )(x)

    # Block 3: Max pooling
    x = layers.MaxPooling2D(pool_size=(2, 1))(x)

    # Block 4: Flatten + Dense
    x = layers.GlobalAveragePooling2D()(x)
    x = layers.Dense(128, activation='relu')(x)
    x = layers.Dropout(0.5)(x)

    outputs = layers.Dense(num_classes, activation='softmax')(x)

    return tf.keras.Model(inputs, outputs)
## 8. Inisialisasi Model

model = build_model(X_train.shape[1:], len(encoder.classes_))
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=LEARNING_RATE,
clipnorm=1.0),
    loss='categorical_crossentropy',
    metrics=['accuracy', tf.keras.metrics.Precision(name='precision'),
tf.keras.metrics.AUC(name='auc')]
)
model.summary()
## 10. Train Model & Callbacks

# Callbacks
callbacks = [
    EarlyStopping(
        monitor='val_loss',
        patience=15,
        restore_best_weights=True
    ),
    ModelCheckpoint(
        os.path.join(save_model_dir, 'best_model.keras'),
        monitor='val_loss',
        save_best_only=True
    ),
    ReduceLROnPlateau(
        monitor='val_loss',
        factor=0.2,
        patience=5,
        min_lr=1e-6
    ),
    CSVLogger(
        os.path.join(save_model_dir, 'training_log.csv'),
```

```
    )
]

# import os
# import pandas as pd
# from tensorflow.keras.models import load_model

# # Path
# model_path = os.path.join(save_model_dir, 'best_model.keras')
# log_path = os.path.join(save_model_dir, 'training_log.csv')

# # Load latest model
# model = load_model(model_path)

# if os.path.exists(log_path):
#     log_df = pd.read_csv(log_path)
#     initial_epoch = len(log_df)
# else:
#     initial_epoch = 0

history = model.fit(
    X_train, y_train_cat,
    validation_data=(X_val, y_val_cat),
    batch_size=BATCH_SIZE,
    # initial_epoch=,
    epochs=EPOCHS,
    callbacks=callbacks,
    verbose=1
)
## 11. Evaluasi & Visualisasi Hasil

# Confusion Matrix
y_pred = model.predict(X_val)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true_classes = np.argmax(y_val_cat, axis=1)

plt.figure(figsize=(8,8))
cm = tf.math.confusion_matrix(y_true_classes, y_pred_classes)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=encoder.classes_,
            yticklabels=encoder.classes_)
plt.title('Confusion Matrix', fontsize=12)
plt.xlabel('Predicted Label')
plt.ylabel('True Label')

nama_file_cm = 'confusion_matrix.png'
path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)
```

```
try:
    plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
    print(f"Confusion matrix berhasil disimpan di: {path_simpan_cm}")
except Exception as e:
    print(f"Gagal menyimpan confusion matrix: {e}")

plt.show()

# Classification Report
from sklearn.metrics import classification_report
report_text = classification_report(y_true_classes, y_pred_classes,
target_names=encoder.classes_)
print("\nClassification Report:")
print(report_text)

nama_file_report = 'classification_report.txt'
path_simpan_report = os.path.join(save_model_dir, nama_file_report)
try:
    with open(path_simpan_report, 'w') as f:
        f.write("Classification Report:\n")
        f.write(report_text)
    print(f"Classification report berhasil disimpan di: {path_simpan_report}")
except Exception as e:
    print(f"Gagal menyimpan classification report: {e}")
# Plot Training History
plt.figure(figsize=(18, 6))

# Accuracy Plot
plt.subplot(1, 3, 1)
plt.plot(history.history['accuracy'], label='Train')
plt.plot(history.history['val_accuracy'], label='Validation')
plt.title('Model Accuracy', fontsize=14)
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend()

# Loss Plot
plt.subplot(1, 3, 2)
plt.plot(history.history['loss'], label='Train')
plt.plot(history.history['val_loss'], label='Validation')
plt.title('Model Loss', fontsize=14)
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend()

# AUC Plot
plt.subplot(1, 3, 3)
plt.plot(history.history['auc'], label='Train')
```

```
plt.plot(history.history['val_auc'], label='Validation')
plt.title('Model AUC', fontsize=14)
plt.ylabel('AUC')
plt.xlabel('Epoch')
plt.legend()

plt.tight_layout()
nama_file_cm = 'plot.png'
path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)

try:
    plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
    print(f"Plot berhasil disimpan di: {path_simpan_cm}")
except Exception as e:
    print(f"Gagal menyimpan plot: {e}")
plt.show()
def visualize_predictions(samples_per_class=3):
    classes = encoder.classes_
    num_classes = len(classes)
    total_samples = num_classes * samples_per_class

    plt.figure(figsize=(25, total_samples*6))

    class_indices = []
    true_labels = np.argmax(y_val_cat, axis=1)

    for cls_idx, cls_name in enumerate(classes):
        indices = np.where(true_labels == cls_idx)[0]
        selected = np.random.choice(indices, samples_per_class, replace=False)
        class_indices.extend(selected)

    # Plotting
    for i, idx in enumerate(class_indices, 1):
        audio = X_val[idx]
        true_label = classes[true_labels[idx]]
        pred_probs = model.predict(np.expand_dims(audio, axis=0),
verbose=0)[0]
        pred_label = classes[np.argmax(pred_probs)]

        # Ekstrak fitur
        mfcc_features = audio[:, :40, 0, 0].T
        spectral_contrast = audio[:, 40:47, 0, 0].T
        chroma = audio[:, 47:59, 0, 0].T

        # Plot waveform
        plt.subplot(total_samples, 5, i*5-4)
        librosa.display.waveshow(audio[:,0,0,0], sr=SR)
        plt.title(f'True: {true_label}\nPred: {pred_label}',
```

```
                fontsize=12, color='green' if true_label == pred_label else 'red')

        # Plot MFCC
        plt.subplot(total_samples, 5, i*5-3)
        librosa.display.specshow(mfcc_features, x_axis='time',
                                 sr=SR, hop_length=HOP_LENGTH, cmap='coolwarm')
        plt.ylabel('MFCC' if i%5 == 1 else '')

        # Plot Spectral Contrast
        plt.subplot(total_samples, 5, i*5-2)
        librosa.display.specshow(spectral_contrast, x_axis='time',
                                 sr=SR, hop_length=HOP_LENGTH, cmap='viridis')
        plt.ylabel('Spectral Contrast' if i%5 == 1 else '')

        # Plot Chroma
        plt.subplot(total_samples, 5, i*5-1)
        librosa.display.specshow(chroma, x_axis='time',
                                 sr=SR, hop_length=HOP_LENGTH, cmap='plasma')
        plt.ylabel('Chroma CQT' if i%5 == 1 else '')

        # Plot probabilitas
        plt.subplot(total_samples, 5, i*5)
        bars = plt.barh(classes, pred_probs, color=['limegreen' if c == true_label
else 'skyblue' for c in classes])
        bars[np.argmax(pred_probs)].set_color('orange')
        plt.xlim(0, 1)
        plt.grid(alpha=0.3)

        if i % samples_per_class == 0 and i != total_samples:
            plt.axhline(y=1.2, color='gray', linestyle='--', linewidth=2, alpha=0.5,
xmin=0.05, xmax=0.95)

    plt.tight_layout()
    nama_file_cm = 'predict_val.png'
    path_simpan_cm = os.path.join(save_model_dir, nama_file_cm)

    try:
        plt.savefig(path_simpan_cm, dpi=300, bbox_inches='tight')
        print(f"Visualisasi prediksi berhasil disimpan di: {path_simpan_cm}")
    except Exception as e:
        print(f"Gagal menyimpan Visualisasi prediksi: {e}")
    plt.show()

print("\nVisualisasi Prediksi per Kelas (3 Sample per Kelas):")
visualize_predictions()
```

DEVELOP WEB: https://github.com/Risvaaa12/Audio-Classify-Question-Generator---ConvLSTM.git

```
# IMPORT LIBRARY
import streamlit as st
import os
import pandas as pd
import json
import numpy as np
import librosa
import noisereduce as nr
import tensorflow as tf
from sklearn.preprocessing import LabelEncoder
import io
from collections import Counter, defaultdict
import re
import openpyxl

# ================================================================
# KONFIGURASI APLIKASI STREAMLIT
# ================================================================
st.set_page_config(
    layout="wide",
    page_title="Harmoni Nusantara - Soal Generator"
)

# ================================================================
# KONSTANTA GLOBAL
# ================================================================
# Pengaturan Audio Processing
SR = 22050
N_FFT = 1024
HOP_LENGTH = 512
N_MFCC = 40
MAX_PAD_LEN = 432
N_SPECTRAL_CONTRAST = 7
N_CHROMA = 12
TOTAL_FEATURES = N_MFCC + N_SPECTRAL_CONTRAST + N_CHROMA
```

```
TARGET_SEGMENT_DURATION_SECONDS = 10

# Kelas Klasifikasi Gamelan
GAMELAN_CLASSES = ['angklung', 'baleganjur', 'gong_gede', 'gong_kebyar',
'semar_pegulingan']

#
================================================
================================
# FUNGSI-FUNGSI HELPER
#
================================================
================================
def parse_excel_questions(file_bytes):
    """Membaca file Excel """
    parsed_questions = []
    try:
        workbook = openpyxl.load_workbook(io.BytesIO(file_bytes))
        sheet = workbook.active
        headers = [cell.value for cell in sheet[1]]

        expected_headers = [
            "Nomor_Level", "ID_Soal_Dalam_Level", "Teks_Pertanyaan",
            "Opsi_A", "Opsi_B", "Opsi_C", "Opsi_D",
"Kunci_Jawaban_Dokumen",
            "Nama_File_Gambar", "Nama_File_Audio_Untuk_Soal"
        ]

        missing_headers = [eh for eh in expected_headers[:7] if eh not in headers]
        if missing_headers:
            st.error(f"Header Excel tidak ditemukan: {', '.join(missing_headers)}.")
            return []

        for row_num, row_values in enumerate(sheet.iter_rows(min_row=2,
values_only=True), start=2):
            if not any(row_values): continue
            row_data = dict(zip(headers, row_values))
            try:
                level_num = int(row_data.get("Nomor_Level"))
                q_id_in_level = int(row_data.get("ID_Soal_Dalam_Level"))
                teks_soal = str(row_data.get("Teks_Pertanyaan", "")).strip()
                kunci = str(row_data.get("Kunci_Jawaban_Dokumen",
")).strip().lower()

                if not teks_soal or not kunci: continue
                if kunci not in ['a', 'b', 'c', 'd']: continue

                parsed_questions.append({
```

```
                'level_number': level_num,
                'id_in_level': q_id_in_level,
                'teks_soal': teks_soal,
                'opsi_jawaban': {
                    'a': str(row_data.get("Opsi_A", "")).strip(), 'b':
str(row_data.get("Opsi_B", "")).strip(),
                    'c': str(row_data.get("Opsi_C", "")).strip(), 'd':
str(row_data.get("Opsi_D", "")).strip()
                },
                'kunci_jawaban_dokumen': kunci,
                'nama_file_gambar': str(row_data.get("Nama_File_Gambar",
"")).strip() or None,
                'nama_file_audio_excel':
str(row_data.get("Nama_File_Audio_Untuk_Soal", "")).strip() or None
            })
        except (ValueError, TypeError): st.caption(f"Baris Excel {row_num}:
Error konversi data, dilewati.")
    except Exception as e: st.error(f"Gagal memproses file Excel: {e}")
    return parsed_questions

def denoise_audio(y, sr):
    """
    Menghilangkan noise dari sinyal audio menggunakan library noisereduce.
    Noise diestimasi dari bagian audio dengan energi terendah.
    """
    try:
        # Tentukan bagian noise berdasarkan energi terendah
        energy = librosa.feature.rms(y=y)
        if energy.size == 0: return y
        percentile_energy = np.percentile(energy, 10)
        noise_frames_indices = np.where(energy.flatten() < percentile_energy)[0]

        # Default noise part jika tidak ada frame noise yang terdeteksi
        noise_part = y[:int(0.1 * len(y))]

        if len(noise_frames_indices) > 0:
            noise_samples_indices =
librosa.frames_to_samples(noise_frames_indices)
            noise_samples_indices = noise_samples_indices[noise_samples_indices
< len(y)]
            if len(noise_samples_indices) > 0:
                noise_part_candidate = y[noise_samples_indices]
                if len(noise_part_candidate) > 0:
                    noise_part = noise_part_candidate

        if len(noise_part) == 0 and len(y) > 0:
            noise_part = y[:int(0.1 * len(y))]
        elif len(y) == 0:
```

```
            return y

        # Parameter untuk noise reduction
        GAMELAN_NR_PARAMS = {
            'stationary': False, 'prop_decrease': 0.5, 'n_fft': 1024,
            'win_length': 256, 'use_tqdm': False
        }

        # Lakukan noise reduction
        if len(noise_part) > 0:
            return nr.reduce_noise(y=y, y_noise=noise_part, sr=sr,
**GAMELAN_NR_PARAMS)
        return y
    except Exception:
        return y

def extract_features_for_streamlit(y, sr):
    """
    Mengekstrak fitur dari sinyal audio untuk input model klasifikasi.
    Termasuk denoising, ekstraksi MFCC, Spectral Contrast, Chroma,
normalisasi, dan padding.
    """
    try:
        y_clean = denoise_audio(y, sr)
        if y_clean is None or len(y_clean) == 0:
            return None

        # Ekstraksi fitur
        mfcc = librosa.feature.mfcc(y=y_clean, sr=sr, n_mfcc=N_MFCC,
n_fft=2048, hop_length=HOP_LENGTH)
        spectral_contrast = librosa.feature.spectral_contrast(y=y_clean, sr=sr,
n_fft=N_FFT, hop_length=HOP_LENGTH)
        chroma = librosa.feature.chroma_cqt(y=y_clean, sr=sr,
hop_length=HOP_LENGTH)

        # Gabungkan semua fitur
        features = np.vstack([mfcc, spectral_contrast, chroma])
        if features.shape[0] != TOTAL_FEATURES:
            return None

        # Normalisasi fitur
        mean, std = np.mean(features, axis=1, keepdims=True), np.std(features,
axis=1, keepdims=True)
        features = (features - mean) / (std + 1e-6)

        # Padding atau pemotongan agar panjangnya sesuai
        if features.shape[1] < MAX_PAD_LEN:
```

```
            features = np.tile(features, int(np.ceil(MAX_PAD_LEN /
features.shape[1])))[:, :MAX_PAD_LEN]
        else:
            features = features[:, :MAX_PAD_LEN]

        # Ubah bentuk array sesuai input model
        return features.T.reshape(MAX_PAD_LEN, TOTAL_FEATURES, 1, 1)
    except Exception:
        return None

@st.cache_resource
def load_gamelan_model():
    """
    Memuat model klasifikasi Gamelan dari file .keras.
    Menggunakan cache Streamlit untuk mencegah pemuatan berulang.
    """
    try:
        model_path = 'ConvLSTM_10s.keras'
        if not os.path.exists(model_path):
            st.error(f"File model '{model_path}' tidak ditemukan.")
            return None
        return tf.keras.models.load_model(model_path)
    except Exception as e:
        st.error(f'Error saat memuat model: {e}")
        return None

def classify_audio_files_streamlit(audio_file_infos,
progress_placeholder=None):
    """
    Mengklasifikasikan daftar file audio yang diunggah.
    Setiap file audio dipotong menjadi segmen-segmen, lalu setiap segmen
diklasifikasikan.
    Hasil akhir adalah kelas yang paling sering muncul dari semua segmen.
    """
    results = []
    if not model:
        st.error("Model klasifikasi tidak berhasil dimuat.")
        return results

    total = len(audio_file_infos)
    if total == 0:
        return results

    seg_len_samples = SR * TARGET_SEGMENT_DURATION_SECONDS
    for i, f_info in enumerate(audio_file_infos):
        fname, abytes, fid = f_info['name'], f_info['bytes'], f_info['id']
        if progress_placeholder:
```

```
        progress_placeholder.progress(i / total, text=f"Proses: {fname}
({i+1}/{total})")

    try:
        # Muat dan resample audio
        y, sr_orig = librosa.load(io.BytesIO(abytes), sr=None)
        if sr_orig != SR:
            y = librosa.resample(y, orig_sr=sr_orig, target_sr=SR)

        # Proses per segmen
        num_samples, seg_preds, pos = len(y), [], 0
        while pos < num_samples:
            seg_y = y[pos:min(pos + seg_len_samples, num_samples)]
            pos += seg_len_samples

            # Lewati segmen yang terlalu pendek
            if len(seg_y) < SR * 0.5:
                if num_samples <= seg_len_samples and len(seg_y) > 0:
                    pass
                elif len(seg_preds) > 0 or len(seg_y) == 0:
                    continue

            # Ekstrak fitur dan prediksi
            feat_seg = extract_features_for_streamlit(seg_y, SR)
            if feat_seg is not None:
                pred_probs = model.predict(np.expand_dims(feat_seg, axis=0),
verbose=0)[0]
                seg_preds.append(encoder.classes_[np.argmax(pred_probs)])

        # Agregasi hasil dari semua segmen
        if seg_preds:
            counts = Counter(seg_preds)
            probs = {cls: float(counts.get(cls, 0)) / len(seg_preds) for cls in
GAMELAN_CLASSES}
            results.append({'filename': fname, 'probabilities': probs, 'id': fid})
        else:
            results.append({'filename': fname, 'probabilities': {}, 'error': 'Tidak
ada segmen audio yang valid.', 'id': fid})
    except Exception as e:
        results.append({'filename': fname, 'probabilities': {}, 'error': str(e), 'id':
fid})

  if progress_placeholder:
      progress_placeholder.progress(1.0, text=f"Validasi {total} file selesai!")
  return results
```

```
#
# ================================================================
# INISIALISASI MODEL, ENCODER, DAN SESSION STATE
#
# ================================================================

# Muat model dan siapkan encoder
model = load_gamelan_model()
encoder = LabelEncoder()
encoder.fit(GAMELAN_CLASSES)

# Inisialisasi session state untuk menyimpan data antar-interaksi
default_session_state = {
    'uploaded_files_info': [],
    'classification_results': [],
    'show_export_dialog': False,
    'uploader_key_counter': 0,
    'uploader_doc_key_counter': 0,
    'show_validation_modal': False,
    'files_being_validated': [],
    'uploaded_question_document_file': None,
    'parsed_document_questions': [],
    'document_parse_error': None,
    'edited_audio_answer_keys': {},
    'audio_to_soal_info_map': {}
}
for key, default_val in default_session_state.items():
    if key not in st.session_state:
        st.session_state[key] = default_val

#
# ================================================================
# FUNGSI-FUNGSI AKSI (CALLBACK)
#
# ================================================================

def open_export_dialog():
    """Membuka dialog ekspor JSON jika file soal sudah diunggah."""
    if not st.session_state.parsed_document_questions:
        st.warning("Unggah dan proses file soal Excel terlebih dahulu.")
    else:
        st.session_state.show_export_dialog = True

def delete_audio_file(file_id):
```

```
    """Menghapus file audio dari session state berdasarkan ID-nya."""
    st.session_state.uploaded_files_info = [f for f in
st.session_state.uploaded_files_info if f['id'] != file_id]
    st.session_state.classification_results = [r for r in
st.session_state.classification_results if r.get('id') != file_id]
    st.session_state.uploader_key_counter += 1
    st.toast("File audio dan hasilnya telah dihapus.", icon="🗑️")

def clear_all_data():
    """Menghapus semua data dari session state untuk memulai dari awal."""
    for k_list in ['uploaded_files_info', 'classification_results',
'parsed_document_questions', 'audio_to_soal_info_map']:
        st.session_state[k_list] = [] if isinstance(st.session_state.get(k_list), list)
else {}
    st.session_state.edited_audio_answer_keys = {}
    for k_none in ['uploaded_question_document_file', 'document_parse_error']:
        st.session_state[k_none] = None
    st.session_state.uploader_key_counter += 1
    st.session_state.uploader_doc_key_counter += 1
    st.toast("Semua data telah dihapus.", icon="♻️")

def process_document(doc_file):
    """Memproses file Excel yang diunggah."""
    st.session_state.parsed_document_questions = []
    st.session_state.document_parse_error = None
    st.session_state.edited_audio_answer_keys = {}
    st.session_state.audio_to_soal_info_map = {}
    st.session_state.uploaded_question_document_file = {"name":
doc_file.name, "id": doc_file.file_id}

    # Validasi ekstensi file
    ext = os.path.splitext(doc_file.name)[1].lower()
    if ext in [".xlsx", ".xls"]:
        parsed = parse_excel_questions(doc_file.getvalue())
        if parsed:
            st.session_state.parsed_document_questions = parsed
            audio_map = {
                q['nama_file_audio_excel']: {'id': q['id_in_level'], 'level':
q['level_number']}
                for q in parsed if q.get('nama_file_audio_excel')
            }
            st.session_state.audio_to_soal_info_map = audio_map
            st.toast(f"Ditemukan {len(audio_map)} soal dengan audio terkait di
Excel.")
        elif not st.session_state.document_parse_error:
            st.warning(f"Tidak ada soal yang berhasil diekstrak dari
'{doc_file.name}'.")
    else:
```

```
        st.session_state.document_parse_error = f"Format file {ext} tidak
didukung. Harap gunakan file Excel (.xlsx)."

#
================================================
================================
# TATA LETAK (LAYOUT) APLIKASI STREAMLIT
#
================================================
================================

# --- Judul Aplikasi ---
st.markdown("<h1 style='text-align: center;'>Harmoni Nusantara - Soal
Generator</h1>", unsafe_allow_html=True)
st.markdown("---")

st.markdown("<h4 style='text-align: center;'>Unggah File Soal Excel
(.xlsx)</h4>", unsafe_allow_html=True)
st.caption("Pastikan Excel memiliki kolom: Nomor_Level,
ID_Soal_Dalam_Level, Teks_Pertanyaan, Opsi_A, Opsi_B, Opsi_C, Opsi_D,
Kunci_Jawaban_Dokumen, Nama_File_Gambar (ops),
Nama_File_Audio_Untuk_Soal (ops).")
uploaded_doc = st.file_uploader(
    "Unggah file Excel",
    type=["xlsx", "xls"],
    key=f"doc_uploader_{st.session_state.uploader_doc_key_counter}"
)

if uploaded_doc:
    current_doc_id = st.session_state.uploaded_question_document_file['id'] if
st.session_state.uploaded_question_document_file else None
    if uploaded_doc.file_id != current_doc_id:
        with st.spinner(f"Memproses '{uploaded_doc.name}'..."):
            process_document(uploaded_doc)
        st.rerun()

if st.session_state.uploaded_question_document_file:
    if st.session_state.parsed_document_questions:
        st.info(f"Berhasil memuat
**{len(st.session_state.parsed_document_questions)}** soal dari dokumen.")
        # with st.expander("Lihat Pratinjau 3 Soal Pertama dari Dokumen"):
        #     st.json(st.session_state.parsed_document_questions[:3])
    elif st.session_state.document_parse_error:
        st.error(f"Error Parsing Dokumen:
{st.session_state.document_parse_error}")

st.markdown("---")
```

```
# --- Bagian Unggah File Audio ---
st.markdown("<h4 style='text-align: center;'>Unggah File Audio
(WAV/MP3/OGG)</h4>", unsafe_allow_html=True)
st.caption("Jika soal memerlukan audio, pastikan nama file audio yang
diunggah sesuai dengan kolom 'Nama_File_Audio_Untuk_Soal' di Excel.")

uploaded_audios = st.file_uploader(
    "Unggah file audio",
    type=["wav", "mp3", "ogg"],
    accept_multiple_files=True,
    key=f"audio_uploader_{st.session_state.uploader_key_counter}"
)

# Tambahkan file audio baru ke session state
if uploaded_audios:
    existing_ids = {f['id'] for f in st.session_state.uploaded_files_info}
    added = False
    soal_info_map = st.session_state.get('audio_to_soal_info_map', {})
    for audio_f in uploaded_audios:
        if audio_f.file_id not in existing_ids:
            st.session_state.uploaded_files_info.append({
                "name": audio_f.name,
                "bytes": audio_f.getvalue(),
                "id": audio_f.file_id,
                "soal_info": soal_info_map.get(audio_f.name)
            })
            added = True
    if added:
        st.rerun()

st.markdown("---")

# --- Tampilan Daftar Audio dan Hasil Validasi ---
col_list_audio, col_hasil_validasi = st.columns(2)

with col_list_audio:
    st.markdown("###### Daftar File Audio Diunggah")
    with st.container(border=True, height=500):
        if not st.session_state.uploaded_files_info:
            st.info("Belum ada file audio yang diunggah.")
        else:
            sorted_files = sorted(
                st.session_state.uploaded_files_info,
                key=lambda x: (
                    x.get('soal_info') is None,
                    x['soal_info'].get('level', float('inf')) if x.get('soal_info') else
float('inf'),
                    x['soal_info'].get('id', float('inf')) if x.get('soal_info') else float('inf'),
```

```
                x['name']
            )
        )
        for f_info in sorted_files:
            # Tampilkan nomor soal dan level dari Excel
            soal_info = f_info.get('soal_info')
            if soal_info:
                no_soal_txt = f"(Soal no. {soal_info['id']}, level
{soal_info['level']})"
            else:
                no_soal_txt = "<span style='color: orange;'>(Audio tidak terkait di
Excel)</span>"

            item_cols = st.columns([0.9, 0.1])
            with item_cols[0]:
                st.markdown(f"<small>{f_info['name']} {no_soal_txt}</small>",
unsafe_allow_html=True)
            with item_cols[1]:
                if st.button("✕", key=f"del_audio_item_{f_info['id']}",
help="Hapus file audio ini"):
                    delete_audio_file(f_info['id'])
                    st.rerun()
            st.divider()

    # Tombol untuk menghapus semua data
    if st.button(
        "🗑️ Hapus Semua Data",
        use_container_width=True,
        key="clear_all_main_btn",
        disabled=not (st.session_state.uploaded_files_info or
st.session_state.uploaded_question_document_file)
    ):
        clear_all_data()
        st.rerun()

with col_hasil_validasi:
    st.markdown("###### Hasil Validasi Audio")
    with st.container(border=True, height=500):
        if not st.session_state.classification_results and
st.session_state.uploaded_files_info:
            st.info("Klik tombol 'Validasi Audio' di bawah untuk memulai proses.")
        elif not st.session_state.classification_results:
            st.info("Belum ada hasil validasi.")
        else:
            audio_map = {f['id']: f for f in st.session_state.uploaded_files_info}
            sorted_results = sorted(
                st.session_state.classification_results,
                key=lambda x: (
```

```
                audio_map.get(x['id'], {}).get('soal_info') is None,
                audio_map.get(x['id'], {}).get('soal_info', {}).get('level', float('inf'))
if audio_map.get(x['id'], {}).get('soal_info') else float('inf'),
                audio_map.get(x['id'], {}).get('soal_info', {}).get('id', float('inf')) if
audio_map.get(x['id'], {}).get('soal_info') else float('inf'),
                x['filename']
            )
        )
        for res in sorted_results:
            f_info_display = audio_map.get(res['id'])
            # Tampilkan nomor soal dan level dari Excel
            no_soal_res_txt = ""
            if f_info_display:
                soal_info = f_info_display.get('soal_info')
                if soal_info:
                    no_soal_res_txt = f"(Soal no. {soal_info['id']}, level
{soal_info['level']})"

            st.markdown(f"<b>{res['filename']}</b>
<small>{no_soal_res_txt}</small>", unsafe_allow_html=True)

            if 'error' in res and res['error']:
                st.error(f"Error: {res['error']}", icon="❗")
            elif 'probabilities' in res and res['probabilities']:
                top_cls = max(res['probabilities'], key=res['probabilities'].get)
                st.write(f"Prediksi: **{top_cls}** (Probabilitas:
{res['probabilities'][top_cls]:.2%})")
                if f_info_display:
                    st.audio(f_info_display['bytes'], format="audio/wav")
            else:
                st.caption("Menunggu validasi atau tidak ada segmen yang valid.")
            st.divider()

    # Tombol untuk memulai validasi audio
    if st.button(
        "▶️ Validasi Audio",
        use_container_width=True,
        key="validate_main_btn",
        disabled=not st.session_state.uploaded_files_info or not model or
st.session_state.get('show_validation_modal')
    ):
        ids_validated = {r['id'] for r in st.session_state.classification_results if
'error' not in r and r.get('probabilities')}
        to_validate = [f for f in st.session_state.uploaded_files_info if f['id'] not in
ids_validated]
        if to_validate:
            st.session_state.files_being_validated = to_validate
            st.session_state.show_validation_modal = True
```

```
            st.rerun()
        elif st.session_state.uploaded_files_info:
            st.toast("Semua file audio sudah berhasil divalidasi.", icon="✅")
        else:
            st.warning("Tidak ada file audio untuk divalidasi.")

# ================================================================
# DIALOG POP-UP
# ================================================================
# --- Dialog Proses Validasi Audio ---
if st.session_state.get('show_validation_modal', False):
    @st.dialog("Proses Validasi Audio")
    def validation_dialog():
        st.info("Validasi audio sedang berlangsung...")
        prog_bar = st.empty()
        prog_bar.progress(0, text="Mempersiapkan...")

        to_validate_now = st.session_state.get('files_being_validated', [])
        if to_validate_now:
            new_results = classify_audio_files_streamlit(to_validate_now,
prog_bar)

            # Update hasil klasifikasi
            current_results = {r['id']: r for r in st.session_state.classification_results}
            for nr in new_results:
                current_results[nr['id']] = nr
            st.session_state.classification_results = list(current_results.values())

            s_count = sum(1 for r in new_results if 'error' not in r or not r['error'])
            if len(new_results) > 0:
                st.toast(f"Validasi selesai. Berhasil: {s_count}/{len(new_results)}.",
icon="📊" if s_count < len(new_results) else "💡")
        else:
            prog_bar.info("Tidak ada file baru untuk divalidasi.")

        # Tutup dialog dan reset state
        st.session_state.show_validation_modal = False
        st.session_state.files_being_validated = []
        st.rerun()

    validation_dialog()

st.markdown("---")
```

```
# --- Tombol Ekspor ---
st.button(
    "⬇ Ekspor Soal ke Format JSON",
    use_container_width=True,
    on_click=open_export_dialog,
    disabled=not st.session_state.parsed_document_questions
)

# --- Dialog Ekspor JSON ---
if st.session_state.get('show_export_dialog', False):
    @st.dialog("Buat File JSON Soal", width="large")
    def export_dialog_final_complete():
        st.markdown("### Pratinjau & Edit Soal Sebelum Ekspor")
        st.caption("Untuk soal audio, Anda dapat mengganti kunci jawaban final
di bawah ini.")
        st.markdown("---")

        doc_qs = st.session_state.parsed_document_questions
        audio_res_map = {res['filename']: res for res in
st.session_state.classification_results if res.get('probabilities')}

        if not doc_qs:
            st.warning("Tidak ada data soal dari Excel untuk ditampilkan.")
            if st.button("Tutup"): st.session_state.show_export_dialog = False;
st.rerun()
            return

        # --- BAGIAN 1: TAMPILKAN UI DETAIL & EDIT ---
        with st.container(height=600, border=True):
            for q_data in doc_qs:
                level_num = q_data.get('level_number')
                q_id = q_data.get('id_in_level')
                q_text = q_data.get('teks_soal', "N/A")
                q_opts = q_data.get('opsi_jawaban', {})
                key_excel = str(q_data.get('kunci_jawaban_dokumen', "")).lower()
                audio_file_excel = q_data.get('nama_file_audio_excel')

                # UID sekarang lebih sederhana karena tidak ada provinsi
                q_uid_tuple = (level_num, q_id)

                with st.container(border=True):
                    st.subheader(f"Level {level_num} - Soal ID {q_id}")
                    st.markdown(f"**Pertanyaan:** {q_text}")
                    st.markdown("**Opsi Jawaban:**")
                    opt_cols = st.columns(2)
                    for i, (k, v) in enumerate(q_opts.items()):
                        opt_cols[i % 2].markdown(f"  **{k.upper()}:**
{v}")
```

```
                st.markdown(f"**Kunci Jawaban dari Excel:**
`{key_excel.upper()}`")

                if audio_file_excel:
                    st.markdown(f"**File Audio Terkait:** `{audio_file_excel}`")
                    pred_display = "_Audio belum divalidasi atau tidak ditemukan_"
                    if audio_file_excel in audio_res_map:
                        probs = audio_res_map[audio_file_excel]['probabilities']
                        if probs:
                            top_cls = max(probs, key=probs.get)
                            pred_display = f"Prediksi Model: **{top_cls}** (Prob:
{probs[top_cls]:.2%})"
                    st.info(pred_display, icon="👹")

                    valid_opts_keys = [k.lower() for k, v in q_opts.items() if v and
v.strip()]
                    key_to_show =
st.session_state.edited_audio_answer_keys.get(q_uid_tuple, key_excel)

                    try:
                        default_index = valid_opts_keys.index(key_to_show) if
key_to_show in valid_opts_keys else 0
                    except ValueError:
                        default_index = 0

                    if valid_opts_keys:
                        edited_key = st.radio(
                            "**Pilih Kunci Jawaban Final:**", valid_opts_keys,
                            index=default_index, key=f"edit_L{level_num}_Q{q_id}",
                            horizontal=True, format_func=lambda x: x.upper()
                        )
                        st.session_state.edited_audio_answer_keys[q_uid_tuple] =
edited_key

        st.markdown("---")

        # --- BAGIAN 2: PROSES DATA UNTUK OUTPUT JSON NESTED ---
        grouped_levels = defaultdict(list)
        for q_data in doc_qs:
            level_num = q_data.get('level_number')
            if level_num is not None:
                grouped_levels[level_num].append(q_data)

        bali_province_object = {
            "nomor_province": 1,
            "nama_province": "Bali",
            "levels_in_province": []
```

```
        }

        for level_num, questions_in_level in sorted(grouped_levels.items()):
            level_object = {"level": level_num, "questions_in_level": []}
            for q_data in questions_in_level:
                q_uid_tuple = (q_data.get('level_number'), q_data.get('id_in_level'))
                final_key =
st.session_state.edited_audio_answer_keys.get(q_uid_tuple,
q_data.get('kunci_jawaban_dokumen'))

                img_file = q_data.get('nama_file_gambar')
                audio_file_excel = q_data.get('nama_file_audio_excel')

                question_object = {
                    "id": q_data.get('id_in_level'),
                    "path_audio": f"/aset/{audio_file_excel}" if audio_file_excel else
None,
                    "path_gambar": f"/aset/{img_file}" if img_file else None,
                    "teks_pertanyaan": q_data.get('teks_soal'),
                    "opsi_jawaban": q_data.get('opsi_jawaban'),
                    "kunci_jawaban_dokumen": final_key
                }
                level_object["questions_in_level"].append(question_object)
            if level_object["questions_in_level"]:
                bali_province_object["levels_in_province"].append(level_object)

        final_json_output = [bali_province_object]

        # --- BAGIAN 3: TOMBOL UNDUH & TUTUP ---
        pretty = st.checkbox("Format JSON agar mudah dibaca (Pretty Print)",
True)
        json_out = json.dumps(final_json_output, indent=4 if pretty else None,
ensure_ascii=False)

        dl_col, cc_col = st.columns(2)
        def close_action(): st.session_state.show_export_dialog = False

        dl_col.download_button(
            "💾 Unduh susunan_soal.json", json_out, "susunan_soal.json",
            "application/json", disabled=not
final_json_output[0]["levels_in_province"],
            use_container_width=True, on_click=close_action
        )
        if cc_col.button("Tutup", use_container_width=True):
            close_action()
            st.rerun()

    export_dialog_final_complete()
```

Lampiran 0.14 Kode Aplikasi *Game*

Quiz Manager: https://github.com/Risvaaa12/SourceCode_Game.git **(**Full Scripts)

```
using UnityEngine;
using UnityEngine.UI;
using TMPro;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using UnityEngine.SceneManagement;

[RequireComponent(typeof(AudioSource))]
public class QuizManager : MonoBehaviour
{
    private LevelData currentLevel;
    private List<QuestionData> questionsForThisRound, missedQuestions = new
List<QuestionData>();
    private bool isRetryRound = false;
    private int currentQuestionIndex, score;
    public float delayAfterAnswer = 2f;
    private bool isQuizOver = false;


    [Header("Referensi UI Narasi")]
    public GameObject narrativePanel;
    public TextMeshProUGUI narrativeText;
    public Image characterImage;
    public Sprite[] characterPoses;
    private int currentNarrativeIndex;

    [Header("Referensi UI Kuis")]
    public GameObject mainQuizUI;
    public GameObject blurBackground;
    public CanvasGroup mainUICanvasGroup;
    public GameObject mediaContainer;
    public TextMeshProUGUI questionText_UI;
    public Image questionImage_UI;
    public Button playAudioButton_UI;
    public Button[] answerButtons;
    public TextMeshProUGUI scoreText;
    public GameObject[] heartIcons;
```

```
[Header("Pengaturan UI Nyawa")]
public Color fullHeartColor = Color.red;
public Color emptyHeartColor = Color.black;

[Header("Pengaturan Warna Feedback")]
public Color defaultButtonColor = Color.white;
public Color correctButtonColor = Color.green;
public Color wrongButtonColor = Color.red;

[Header("Referensi UI Panel")]
public GameObject rewardPanel;
public GameObject failedPanel;

[Header("Referensi UI Hadiah")]
public Image rewardIcon;
public TextMeshProUGUI rewardText;
public Button claimButton;
public Button backToMapButton;

[Header("Efek Suara")]
public AudioClip correctSound;
public AudioClip wrongSound;
public AudioClip winSound;
public AudioClip loseSound;
private AudioSource sfxAudioSource;
public AudioSource quizSceneBGM;
public AudioSource questionAudioSource;

[Header("Referensi Animator")]
public Animator rewardPanelAnimator;
public Animator failedPanelAnimator;



void Start()
{
   sfxAudioSource = GetComponent<AudioSource>();
   currentLevel = GameManager.Instance.currentLevelToPlay;

   if (currentLevel == null) { GoToMapScene(); return; }

   mainQuizUI.SetActive(false);
   narrativePanel.SetActive(false);
   rewardPanel.SetActive(false);
   failedPanel.SetActive(false);

   GameManager.Instance.OnStartLevel();
```

```
        if (currentLevel.narrativeSentences != null &&
currentLevel.narrativeSentences.Length > 0)
        {
            StartNarrative();
        }
        else
        {
            StartQuiz();
        }
    }



    void PlayQuestionAudio(AudioClip clip)
    {
        if (clip != null && questionAudioSource != null)
        {
            questionAudioSource.clip = clip;
            questionAudioSource.Play();
        }
    }

    void StartNarrative()
    {
        currentNarrativeIndex = 0;
        narrativePanel.SetActive(true);

        if (characterPoses != null && characterPoses.Length > 0)
        {
            characterImage.sprite = characterPoses[0];
        }

        Button narrativeButton = narrativePanel.GetComponent<Button>();
        narrativeButton.onClick.RemoveAllListeners();
        narrativeButton.onClick.AddListener(ShowNextNarrativeSentence);

        ShowNextNarrativeSentence();
    }

    void ShowNextNarrativeSentence()
    {
        if (currentNarrativeIndex < currentLevel.narrativeSentences.Length)
        {
            narrativeText.text =
currentLevel.narrativeSentences[currentNarrativeIndex];

            if (characterPoses != null && characterPoses.Length > 0)
```

```
            {

                int poseIndex = currentNarrativeIndex % characterPoses.Length;
                characterImage.sprite = characterPoses[poseIndex];
            }

            currentNarrativeIndex++;
        }
        else
        {
            EndNarrative();
        }
    }

    void EndNarrative()
    {
        narrativePanel.SetActive(false);
        StartQuiz();
    }
    void StartQuiz()
    {
        mainQuizUI.SetActive(true);
        if (quizSceneBGM != null && !quizSceneBGM.isPlaying)
        {
            quizSceneBGM.Play();
        }
        InitializeQuiz();
    }

    void InitializeQuiz()
    {
        missedQuestions.Clear();
        isRetryRound = false;
        score = 0;
        currentQuestionIndex = 0;
        isQuizOver = false;
        questionsForThisRound                                                    =
currentLevel.questionPool.Take(currentLevel.numberOfQuestionsToAsk).ToLi
st();
        UpdateLivesUI();
        UpdateScoreUI();
        ShowNextQuestion();
    }

    void ShowEndOfQuizPopup(GameObject panelToShow)
    {
        blurBackground.SetActive(true);
        mainUICanvasGroup.interactable = false;
```

```
        panelToShow.SetActive(true);
    }

    void ShowNextQuestion()
    {
        if (currentQuestionIndex >= questionsForThisRound.Count)
        {
            if (missedQuestions.Count > 0) { isRetryRound = true;
                questionsForThisRound = new
List<QuestionData>(missedQuestions);
                missedQuestions.Clear(); currentQuestionIndex = 0;
ShowNextQuestion();
            } else
            {
                QuizWon();
            }
            return; }
        DisplayQuestion(questionsForThisRound[currentQuestionIndex]);
    }

    void DisplayQuestion(QuestionData q)
    {
        questionText_UI.text = q.questionText;

        bool hasMedia = q.questionImage != null || q.questionAudio != null;
        mediaContainer.SetActive(hasMedia);

        if (hasMedia)
        {
            bool hasImage = q.questionImage != null;
            questionImage_UI.gameObject.SetActive(hasImage);
            playAudioButton_UI.gameObject.SetActive(!hasImage);

            if (hasImage)
            {
                questionImage_UI.sprite = q.questionImage;
            }
            else
            {
                if (quizSceneBGM != null && quizSceneBGM.isPlaying)
                {
                    quizSceneBGM.Pause();
                }

                playAudioButton_UI.onClick.RemoveAllListeners();
                playAudioButton_UI.onClick.AddListener(() =>
PlayQuestionAudio(q.questionAudio));
                PlayQuestionAudio(q.questionAudio);
```

```
            }
        }

        RectTransform textRect = questionText_UI.GetComponent<RectTransform>();

        if (hasMedia)
        {
            textRect.anchorMin = new Vector2(0.5f, 0);
            textRect.anchorMax = new Vector2(0.5f, 0);
            textRect.pivot = new Vector2(0.5f, 0);
            textRect.sizeDelta = new Vector2(800, 250);
            textRect.anchoredPosition = new Vector2(0, 50);
        }
        else
        {
            textRect.anchorMin = new Vector2(0.5f, 0.5f);
            textRect.anchorMax = new Vector2(0.5f, 0.5f);
            textRect.pivot = new Vector2(0.5f, 0.5f);
            textRect.sizeDelta = new Vector2(800, 700);
            textRect.anchoredPosition = new Vector2(0, 0);
        }

        for (int i = 0; i < answerButtons.Length; i++)
        {
            answerButtons[i].interactable = true;
            answerButtons[i].GetComponent<Image>().color = defaultButtonColor;
            answerButtons[i].GetComponentInChildren<TextMeshProUGUI>().text = q.answers[i];
            int answerIndex = i;
            answerButtons[i].onClick.RemoveAllListeners();
            answerButtons[i].onClick.AddListener(() => OnAnswerSelected(answerIndex));
        }
    }

    void OnAnswerSelected(int selectedIndex)
    {
        if (isQuizOver) return;

        if (questionAudioSource != null && questionAudioSource.isPlaying)
        {
            questionAudioSource.Stop();
        }

        foreach (var btn in answerButtons) { btn.interactable = false; }
```

```
        QuestionData currentQuestion = questionsForThisRound[currentQuestionIndex];
        bool isCorrect = (selectedIndex == currentQuestion.correctAnswerIndex);

        if (isCorrect)
        {
            answerButtons[selectedIndex].GetComponent<Image>().color = correctButtonColor;
            PlaySfx(correctSound);

            if (!isRetryRound)
            {
                score++;
                UpdateScoreUI();
            }
        }
        else
        {
            answerButtons[selectedIndex].GetComponent<Image>().color = wrongButtonColor;

            if (!isRetryRound)
            {
                missedQuestions.Add(currentQuestion);
            }

            GameManager.Instance.LoseLife();
            PlaySfx(wrongSound);
            UpdateLivesUI();
        }

        if (GameManager.Instance.playerLives <= 0)
        {
            StartCoroutine(DelayedGameOver());
            return;
        }
        else
        {
            currentQuestionIndex++;
            StartCoroutine(ProceedToNextQuestion());
        }
    }

    IEnumerator DelayedGameOver()
    {
        yield return new WaitForSeconds(delayAfterAnswer);
        GameOver();
    }
```

```
    IEnumerator ProceedToNextQuestion()
    {
        if (questionAudioSource != null && questionAudioSource.isPlaying)
        {
            yield return new WaitWhile(() => questionAudioSource.isPlaying);
        }
        yield return new WaitForSeconds(delayAfterAnswer);

        if (quizSceneBGM != null && !quizSceneBGM.isPlaying)
        {
            int nextIndex = currentQuestionIndex;
            bool nextQuestionIsAudio = (nextIndex < questionsForThisRound.Count
&& questionsForThisRound[nextIndex].questionAudio != null);

            if (!nextQuestionIsAudio)
            {
                quizSceneBGM.UnPause();
            }
        }

        ShowNextQuestion();
    }

    void UpdateLivesUI()
    {
        for (int i = 0; i < heartIcons.Length; i++)
        {
            Image heartImage = heartIcons[i].GetComponent<Image>();

            if (heartImage != null)
            {
                if (i < GameManager.Instance.playerLives)
                {
                    heartImage.color = fullHeartColor;
                }
                else
                {
                    heartImage.color = emptyHeartColor;
                }
            }
        }
    }
    void UpdateScoreUI() { scoreText.text = $"{score} /
{questionsForThisRound.Count} Pertanyaan"; }
    void GameOver()
    {
```

```
        if (isQuizOver) return;
        isQuizOver = true;

        GameManager.Instance.PlayerFailedLevel();

        PlaySfx(loseSound);
        blurBackground.SetActive(true);
        mainUICanvasGroup.interactable = false;
        if (failedPanelAnimator != null)
        {
            failedPanel.SetActive(true);
            failedPanelAnimator.SetTrigger("Show");
        }
        else
        {
            failedPanel.SetActive(true);
        }
    }
    void QuizWon()
    {
        if (isQuizOver) return;
        isQuizOver = true;

        PlaySfx(winSound);

        if(rewardPanel != null)
        {
            if (rewardText != null) rewardText.text = $"Good Job!\nKamu
mendapatkan {currentLevel.reward.instrumentName}!";
            if (rewardIcon != null) rewardIcon.sprite =
currentLevel.reward.instrumentIcon;

            blurBackground.SetActive(true);
            mainUICanvasGroup.interactable = false;
            if (rewardPanelAnimator != null)
            {
                rewardPanel.SetActive(true);
                rewardPanelAnimator.SetTrigger("Show");
            }
            else
            {
                rewardPanel.SetActive(true);
            }
        }
    }
    void PlaySfx(AudioClip clip)
    {
        if (clip != null && sfxAudioSource != null)
```

```
        {
            sfxAudioSource.PlayOneShot(clip);
        }
    }
    private void ProcessRewardClaim()
    {
        GameManager.Instance.LevelCompleted();
    }

    public void PlayWonInstrument()
    {
        if (currentLevel == null || currentLevel.reward == null) return;

        GameManager.Instance.LevelCompleted();

        GameManager.Instance.selectedInstrumentToPlay = currentLevel.reward;

        LoadingManager.Instance.LoadScene("AlatMusik");
    }
    public void ClaimRewardAndGoToMap()
    {
        ProcessRewardClaim();
        LoadingManager.Instance.LoadScene("stage_level");
    }

    public void ClaimRewardAndGoToHome()
    {
        ProcessRewardClaim();
        LoadingManager.Instance.LoadScene("HomeScene");
    }

    public void GoToMapScene()
    {
        if (quizSceneBGM != null) quizSceneBGM.Stop();
        LoadingManager.Instance.LoadScene("stage_level");
    }
    public void GoToHome()
    {
        if (quizSceneBGM != null) quizSceneBGM.Stop();
        LoadingManager.Instance.LoadScene("HomeScene");
    }
}
```