Pengembangan Color Guide Generator Berbasis Algoritma K-Means Clustering Dan Decision Tree Untuk Pembuatan Design Token Json

Yanwari Rahmad Mujianto[1*], Eko Purwanto[2], Sopingi[3]

[1]Program Studi Sistem Informasi, Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[1*]220101079@mhs.udb.ac.id

[2]Program Studi Sistem Informasi, Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[2]eko_purwanto@udb.ac.id

[3]Program Studi Sistem Informasi, Fakultas Ilmu Komputer
Universitas Duta Bangsa Surakarta
[3]sopingi@udb.ac.id

Abstrak— **Penentuan palet warna secara manual dari gambar referensi desain merupakan proses yang tidak efisien dan rentan inkonsistensi. Penelitian ini mengembangkan sistem *Color Guide Generator* berbasis Python yang mengekstraksi warna dominan dari gambar antarmuka menggunakan K-Means *Clustering*, mengklasifikasikannya ke dalam peran desain terstandar menggunakan *Decision Tree*, dan menghasilkan *design token* dalam format JSON serta elemen pendukung seperti CSS *Variables* dan konfigurasi Tailwind CSS. Model *Decision Tree* dilatih dari dataset HSL yang diekstrak dari tiga *design system* publik dan mencapai F1-*score* 0,98 untuk mode terang dan 0,92 untuk mode gelap. Validasi melalui *User Acceptance Testing* (UAT) terhadap 5 responden menghasilkan rata-rata skor 4,52 dari 5. Sistem ini terbukti mengotomatiskan proses penentuan warna yang sebelumnya dilakukan secara manual menjadi alur yang konsisten dan siap diintegrasikan ke dalam pengembangan antarmuka web.**
Kata kunci— Color Guide, K-Means Clustering, Decision Tree, Design Token, Antarmuka Pengguna.

Abstract— **Manual color palette extraction from design reference images is an inefficient and inconsistency-prone process. This study develops a Python-based *Color Guide Generator* system that extracts dominant colors from user interface images using K-Means Clustering, classifies them into standardized design roles using a Decision Tree model, and generates design tokens in JSON, CSS Variables, and Tailwind CSS configuration formats. The Decision Tree model was trained on an HSL dataset derived from three public design systems, achieving an F1-score of 0.98 for light mode and 0.92 for dark mode. User Acceptance Testing (UAT) with 5 respondents yielded an average score of 4,52 out of 5. The system successfully automates the previously manual color selection process into a consistent, web-ready workflow.**
Keywords— Color Guide, K-Means Clustering, Decision Tree, Design Token, User Interface.

I. PENDAHULUAN

Warna adalah elemen visual yang strategis dalam desain antarmuka digital. Pemilihan warna yang tepat terbukti meningkatkan persepsi estetika, keterbacaan konten, dan rasa percaya pengguna terhadap produk digital [1] Namun menjaga konsistensi warna dalam skala proyek yang besar bukan perkara sederhana, desainer kerap harus mengambil sampel warna secara manual dari gambar referensi, mencatat kode warna, lalu menerapkannya satu per satu ke seluruh komponen antarmuka.

Solusi yang kini banyak diadopsi adalah penggunaan design token dalam representasi nilai visual seperti warna yang disimpan sebagai variabel terstandar dalam format JSON. Pendekatan ini terbukti meningkatkan konsistensi visual sekaligus efisiensi kerja tim pengembangan antarmuka [2]. Namun proses pembentukan token dari gambar referensi masih dilakukan secara manual, sehingga tetap rentan terhadap inkonsistensi dan subjektivitas.

Algoritma K-Means *Clustering* telah terbukti efektif untuk mengekstraksi warna dominan dari citra digital. K-Means mengelompokkan piksel gambar berdasarkan kedekatan nilai warna di ruang RGB menggunakan *Euclidean Distance*, menghasilkan centroid yang merepresentasikan warna dominan tiap kelompok [3]. Implementasinya menggunakan Python dan OpenCV telah terbukti layak untuk keperluan ekstraksi warna otomatis pada objek visual digital [4].

Namun ekstraksi warna saja belum cukup. Penelitian [4] misalnya, hanya menghasilkan palet warna visual statis dari citra digital tanpa

mengklasifikasikan warna ke dalam peran desain yang bermakna seperti primary, background, atau surface, sehingga hasilnya tidak dapat langsung digunakan sebagai color token yang fungsional.

Padahal klasifikasi peran inilah yang menjadi syarat agar hasil ekstraksi dapat langsung diimplementasikan. Penelitian [5] menunjukkan bahwa algoritma klasifikasi berbasis data mampu menghasilkan segmentasi warna yang lebih akurat dan konsisten dibandingkan penilaian manual, sehingga memperkuat perlunya tahap klasifikasi otomatis dalam proses ekstraksi warna.

Persoalan serupa juga terlihat pada penelitian [6], yang meskipun berhasil mengekstraksi palet warna dominan dengan kualitas visual tinggi, keluarannya masih diarahkan untuk kompresi citra dan belum mencakup klasifikasi peran warna untuk kebutuhan design token. Di sinilah algoritma Decision Tree berperan sebagai model supervised learning yang mampu mengklasifikasikan nilai warna HSL ke dalam peran desain secara otomatis dan konsisten.

Penelitian ini mengusulkan sistem Color Guide Generator yang mengintegrasikan deteksi area antarmuka berbasis OpenCV, ekstraksi warna dominan dengan K-Means Clustering, dan klasifikasi peran desain menggunakan Decision Tree. Luarannya berupa design token siap pakai dalam format JSON, CSS Variables, dan konfigurasi Tailwind CSS, menggantikan proses manual yang selama ini tidak efisien dengan alur otomatis yang konsisten dan siap diintegrasikan ke workflow pengembangan antarmuka web.

II. METODOLOGI PENELITIAN

A. *K-Means Clustering*

K-Means Clustering adalah algoritma unsupervised learning yang mengelompokkan data ke dalam K kelompok berdasarkan kedekatan karakteristik. Dalam penelitian ini, algoritma digunakan untuk mengelompokkan piksel gambar antarmuka pada ruang warna RGB menjadi enam warna dominan (k=6). Proses K-Means berlangsung secara iteratif yang artinya centroid awal dipilih secara acak, setiap piksel ditetapkan ke centroid terdekat, lalu centroid diperbarui hingga konvergen [6]

Penetapan piksel ke centroid menggunakan perhitungan *Euclidean Distance* pada ruang warna RGB sebagai berikut:

$$d(p, c) = \sqrt{[(R_p - R_c)^2 + (G_p - G_c)^2 + (B_p - B_c)^2]}$$

p adalah piksel dengan nilai (R_p, G_p, B_p) dan c adalah centroid dengan nilai (R_c, G_c, B_c). Piksel ditetapkan ke centroid dengan nilai d terkecil. Pembaruan centroid dilakukan dengan menghitung rata-rata seluruh piksel dalam kelompok tersebut [3].

B. *Deteksi Area UI Berbasis OpenCV*

Sebelum ekstraksi warna dilakukan, sistem memisahkan area antarmuka (UI) dari elemen foto atau non-UI menggunakan analisis blok berbasis OpenCV yang merupakan pustaka pengolahan citra digital yang mendukung berbagai operasi analisis visual [7]. Sebuah blok diklasifikasikan sebagai foto hanya jika keempat kriteria terpenuhi secara bersamaan :

1. *Edge Density* (Canny) > 8%
2. Standar deviasi piksel > 45
3. Variasi warna RGB > 2.000
4. Keragaman warna > 25 warna per blok

Kriteria keragaman warna unik menjadi pembeda utama antara elemen foto dan teks atau komponen UI, karena foto mengandung 30–60 warna unik per blok sedangkan komponen UI umumnya hanya mengandung 5–24 warna unik. Area yang terdeteksi sebagai foto dieliminasi menggunakan morphological closing dan penyaringan komponen konektif.

C. *Decision Tree*

Decision Tree adalah algoritma supervised learning yang mengklasifikasikan data berdasarkan serangkaian aturan percabangan pada fitur input [8]. Dalam penelitian ini, setiap warna dominan hasil K-Means dikonversi dari RGB ke ruang warna HSL, kemudian diklasifikasikan ke dalam peran desain (primary, secondary, background, surface, text, border, accent) menggunakan model yang dilatih dari dataset tiga design system publik: Material Design 3, Tailwind CSS v3, dan Ant Design v5.

Pemilihan fitur split pada setiap node pohon menggunakan kriteria Gini Impurity, yang mengukur tingkat kemurnian sebuah node. Semakin rendah nilai Gini, semakin murni pengelompokan kelas pada node tersebut. Rumus Gini Impurity adalah:

$$\text{Gini}(t) = 1 - \Sigma\,[p(i|t)]^2$$

$$i = 1, 2, ..., c$$

p(i|t) adalah proporsi sampel kelas i pada node t, dan c adalah jumlah kelas. Nilai Gini = 0 menunjukkan node yang sepenuhnya murni (semua sampel satu kelas).

Model dilatih menggunakan DecisionTreeClassifier(max_depth=10) dari scikit-learn dengan split data 80:20 secara stratifikasi. Deteksi tema light atau dark mode dilakukan otomatis berdasarkan nilai weighted lightness warna dominan, jika median tertimbang L > 50 maka terdeteksi sebagai light mode, kalau sebaliknya berarti dark mode. Lalu model .pkl yang berbeda dimuat untuk masing-masing tema.

D. Evaluasi Model

Evaluasi performa model Decision Tree dilakukan menggunakan tiga metrik klasifikasi yaitu Precision, Recall, dan F1-score [9]. Ketiga metrik dihitung berdasarkan nilai True Positive (TP), False Positive (FP), dan False Negative (FN) dari hasil prediksi peran warna.

$$\textit{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

Precision adalah ukuran yang menunjukkan seberapa tepat hasil prediksi yang dihasilkan oleh model.

$$\textit{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

Recall menggambarkan kemampuan model dalam menemukan seluruh peran warna yang benar dari data yang tersedia..

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

F1-score merupakan ukuran yang menggabungkan nilai Precision dan Recall untuk memberikan penilaian terhadap performa model secara keseluruhan.

Kualitas pengelompokan K-Means dievaluasi menggunakan Silhouette Score, yang mengukur seberapa mirip sebuah piksel dengan kelompoknya sendiri dibandingkan kelompok lain, dengan rentang nilai −1 hingga +1. Nilai mendekati +1 menunjukkan pengelompokan yang baik dan terdefinisi jelas.

E. Metode Pengumpulan Data

Penelitian ini menggunakan dua jenis data. Pertama, data pelatihan model Decision Tree yang diperoleh dengan mengekstraksi nilai warna HEX dari dokumentasi resmi tiga *design system* publik yang menyediakan token warna dan peran semantik secara terstruktur [10], yaitu:

1. *Material* Design 3 : diambil token warna peran semantik (primary, secondary, background, dst.) dari m3.material.io/styles/color/roles
2. Tailwind CSS v3 : diambil seluruh palet warna per skala kecerahan dari v3.tailwindcss.com/docs/customizing-colors
3. Ant Design v5 : diambil token warna komponen dan semantik dari ant.design/docs/spec/colors

Setiap nilai warna dalam format HEX dikonversi ke ruang warna HSL menggunakan pustaka *colorsys* Python untuk memperoleh representasi Hue, Saturation, dan Lightness yang digunakan sebagai fitur masukan model Decision Tree [11]. dan diberi label peran desain sesuai konteks sumbernya, menghasilkan dataset 406 baris untuk mode terang dan 119 baris untuk mode gelap. Kedua, data gambar uji berupa screenshot antarmuka digital dalam format JPG/PNG (maksimal 5 MB) yang umum digunakan sebagai format penyimpanan citra digital [12], diambil dari dokumentasi pribadi

milik peneliti yang nantinya digunakan untuk menguji pipeline sistem secara end-to-end.

III. HASIL DAN PEMBAHASAN

A. *Pembangunan Dataset dan Pelatihan Model*

Dataset pelatihan dibangun secara manual dengan mengekstraksi nilai warna langsung dari dokumentasi resmi tiga *design system* publik. Setiap sumber memiliki penamaan peran warna yang berbeda-beda, sehingga diperlukan pemetaan ulang ke tujuh peran desain yang seragam: *primary*, *secondary*, *background*, *surface*, *text*, *border*, dan *accent*. Pemetaan dilakukan sebagai berikut :

1. Material Design 3 : sudah mendefinisikan peran warna secara eksplisit seperti Primary, On-Primary, Secondary, Background, Surface. Pemetaan langsung ke label standar tanpa banyak perubahan.
2. Tailwind CSS v3 : Menyediakan palet skala kecerahan (50–950) per warna. Skala gelap (700–950) dipetakan ke *text*, skala tengah (400–600) ke *primary/secondary/accent*, dan skala terang (50–200) ke *background/surface*.
3. Ant Design v5 : menggunakan sistem token semantik seperti *colorPrimary*, *colorBgContainer*, *colorBorder*, *colorText*. Setiap token dipetakan ke label peran yang paling sesuai secara semantik.

Setiap nilai warna HEX dari ketiga sumber dikonversi ke ruang warna HSL menggunakan pustaka colorsys bawaan Python. Konversi ini menghasilkan tiga fitur numerik yaitu *Hue* (H), *Saturation* (S), dan *Lightness* (L) yang digunakan sebagai fitur input model. Hasil konversi disimpan dalam dua file CSV terpisah berdasarkan tema, yaitu color_roles_light.csv (406 baris) dan color_roles_dark.csv (119 baris), masing-masing dengan kolom H, S, L, dan role sebagai label kelas.

Model Decision Tree dilatih secara terpisah untuk setiap tema menggunakan DecisionTreeClassifier (max_depth=10) dari scikit-learn. Model yang telah dilatih disimpan dalam format .pkl menggunakan pustaka pickle Python, menghasilkan dua file yaitu decision_tree_light.pkl & decision_tree_dark.pkl sehingga dapat dimuat kembali saat pipeline berjalan tanpa perlu melatih ulang.

B. *Evaluasi Model Decision Tree*

Evaluasi model dilakukan menggunakan program Python dengan library scikit-learn. Model .pkl dimuat kembali, diuji terhadap data uji, lalu menghasilkan laporan evaluasi berupa nilai *precision*, *recall*, dan F1-*score* per kelas peran warna.

Hasil evaluasi yang dijalankan menggunakan program Python tersebut menghasilkan laporan sebagai berikut :

Tabel 1. Hasil Evaluasi Decision Tree (Light Mode)

Role	Precision	Recall	F1-Score	Support
Accent	1.00	1.00	1.00	15
Background	0.88	1.00	0.94	15
Border	1.00	1.00	1.00	15
Primary	1.00	1.00	1.00	15
Secondary	1.00	1.00	1.00	15
Surface	1.00	0.87	0.93	15
Text	1.00	1.00	1.00	15
Weight Avg	0.98	0.98	0.98	105

Tabel 2. Hasil Evaluasi Decision Tree (Dark Mode)

Role	Precision	Recall	F1-Score	Support
Accent	1.00	0.75	0.86	4
Background	1.00	1.00	1.00	4
Border	0.75	1.00	0.86	3
Primary	1.00	1.00	1.00	3
Secondary	0.50	1.00	0.67	1
Surface	1.00	0.75	0.86	4
Text	1.00	1.00	1.00	5
Weight Avg	0.95	0.92	0.92	24

Berdasarkan hasil di atas, model mode terang mencapai F1-*score* 0,98 dan model mode gelap mencapai 0,92. Keduanya melampaui target kelayakan 0,85 yang telah ditetapkan. Nilai F1 yang lebih rendah pada kelas *secondary* di mode gelap wajar terjadi karena jumlah referensi warna *secondary* pada *design system* mode gelap memang lebih sedikit dibanding kelas lainnya.

C. *Evaluasi K-Means Clustering*

Kualitas pengelompokan warna oleh K-Means dievaluasi menggunakan dua metrik yaitu *Inertia* dan *Silhouette Score*. Evaluasi dijalankan langsung setelah proses K-Means selesai menggunakan library scikit-learn. Hasil evaluasi ditampilkan pada Tabel 3.

Tabel 3. Hasil Evaluasi K-Means Clustering

Metrik	Nilai	Keterangan
Inertia	10.686.537,75	Total jarak piksel ke centroid cluster masing-masing
Silhouette Score	0,8656	Tingkat keterpisahan antar cluster (rentang −1 hingga +1)

Inertia mengukur seberapa rapat piksel berkumpul di dalam cluster-nya masing-masing yang semakin kecil nilainya, semakin rapat. Nilai inertia sebesar 10.686.537,75 tergolong wajar mengingat jumlah piksel yang diproses sangat besar (sampel 50.000 piksel dari 2 juta piksel area UI) dengan rentang nilai RGB 0–255.

Silhouette Score merupakan metode evaluasi internal yang digunakan untuk mengukur kualitas hasil clustering berdasarkan tingkat kemiripan suatu data terhadap cluster-nya sendiri dibandingkan dengan cluster lain, dengan rentang nilai −1 hingga +1 [13]. Nilai 0,8656 yang dihasilkan tergolong sangat baik yaitu mendekati +1 menunjukkan bahwa 6 cluster warna yang terbentuk memiliki batas yang jelas dan tidak tumpang tindih. Ini berarti K-Means berhasil mengelompokkan warna-warna dominan antarmuka BookWalking secara akurat dan representatif.

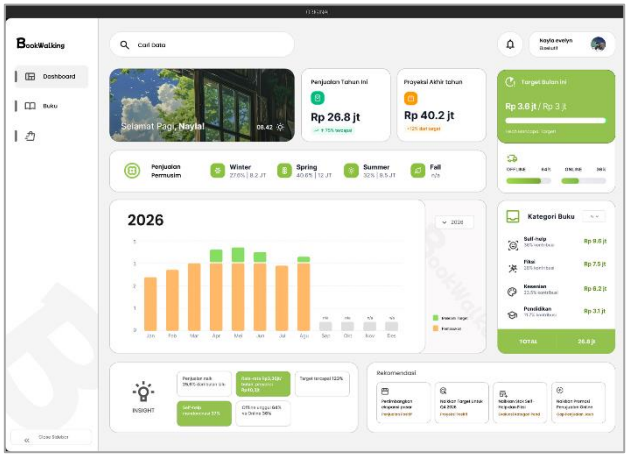
D. *Hasil Deteksi Area dan Ekstraksi Warna Dominan*

Pengujian dilakukan menggunakan gambar antarmuka dashboard aplikasi BookWalking (test01.jpg) yang diambil dari dokumentasi pribadi peneliti. Proses berjalan dalam dua tahap menggunakan Python dengan library OpenCV dan scikit-learn.

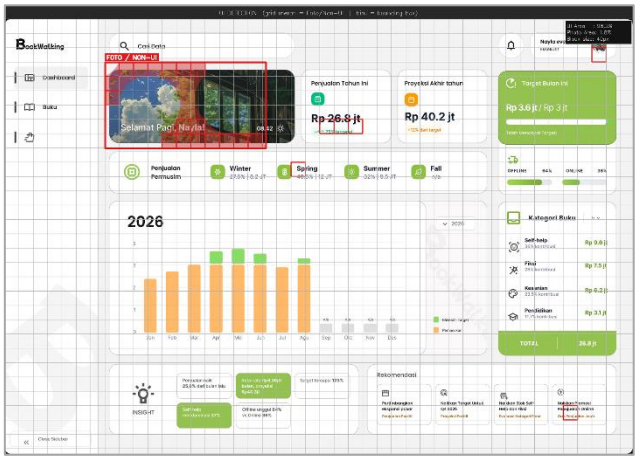
1. Deteksi UI

Sistem memindai gambar blok per blok (40×40 piksel) menggunakan OpenCV untuk memisahkan area antarmuka dari elemen foto. Sebuah blok dinyatakan sebagai foto hanya jika memenuhi tiga kriteria sekaligus: standar deviasi piksel di atas 60, variasi warna RGB di atas 3.000, dan jumlah warna unik per blok di atas 30.

Kriteria ketiga menjadi pembeda utama, komponen UI seperti tombol dan latar hanya memiliki 2–15 warna unik per blok, sedangkan foto biasanya mengandung 30 warna unik atau lebih. Hasil deteksi ditunjukkan pada komparasi pada Gambar 1 dan Gambar 2.



Gambar 1. Antarmuka Sebelum Proses Deteksi



Gambar 2. Hasil Deteksi Area UI

Dari hasil pemindaian, area UI terdeteksi sebesar 98,2% (2.094.204 piksel) dan area foto yang dieliminasi hanya 1,8% (38.121 piksel) berupa foto sampul buku di bagian kiri atas antarmuka. Sistem juga menghitung nilai kecerahan rata-rata dari piksel UI dan mendapatkan nilai 91,17, sehingga gambar terdeteksi otomatis sebagai *light mode*.

2. Ekstraksi Warna Dominan

Ekstraksi warna dominan merupakan proses untuk memperoleh sekumpulan warna yang paling merepresentasikan karakteristik visual suatu citra. Pada penelitian ini, proses tersebut dilakukan menggunakan algoritma K-Means Clustering dengan k=6 [14]. Dari piksel area UI yang tersisa, scikit-learn menjalankan K-Means Clustering untuk menemukan 6 warna yang paling mewakili tampilan antarmuka.. Hasil ekstraksi ditampilkan pada Tabel 4.

Tabel 4. Hasil Ekstraksi Warna Dominan

#	HEX	H	S (%)	L (%)	Coverage	Warna
1	#fbfbfb	0.0	0.0	98.4	84.4%	Putih bersih
2	#92c14e	84.5	48.1	53.1	5.7%	Hijau
3	#feb866	32.4	98.7	69.8	2.6%	Oranye muda
4	#c7c9c3	80.0	5.3	77.7	2.5%	Abu-abu terang
5	#202723	145.7	9.9	13.9	2.5%	Hitam kehijauan
6	#5d7475	182.5	11.4	41.2	2.2%	Abu-abu biru

Warna pertama (#fbfbfb) mendominasi dengan coverage 84,4%, mencerminkan latar putih bersih yang mendominasi tampilan BookWalking. Lima warna sisanya masing-masing di bawah 6% namun merepresentasikan elemen-elemen penting seperti warna hijau aksen, oranye, abu-abu, teks, dan warna pendukung. Keenam warna ini selanjutnya diteruskan ke tahap klasifikasi peran desain.

E. Hasil klasifikasi warna dan generated token

Keenam warna hasil K-Means diklasifikasikan ke dalam peran desain menggunakan model *Decision Tree*. Setiap warna dimasukkan dalam format nilai HSL, lalu model menentukan perannya secara otomatis. Sistem juga menangani konflik apabila dua warna mendapatkan prediksi peran yang sama, warna dengan *coverage* lebih kecil dialihkan ke peran lain yang paling sesuai (*conflict resolution*). Hasil klasifikasi ditampilkan pada Tabel 5.

Tabel 5. Hasil Klasifikasi Peran Warna

Role	HEX	Coverage	Confidence	Warna
background	#fbfbfb	84.6%	0.93	Putih bersih
accent	#92c14e	5.9%	1.00	Hijau
border	#c7c9c5	2.6%	1.00	Abu-abu terang
primary *	#feb867	2.4%	1.00	Oranye muda
text	#202723	2.4%	1.00	Hitam kehijauan
secondary	#5d7377	2.1%	1.00	Abu-abu biru

Setelah seluruh warna berhasil diklasifikasikan, sistem secara otomatis menghasilkan tiga format *design token* yang tersimpan di folder output. Format JSON menjadi output utama karena paling lengkap dan fleksibel [15].

Setiap warna menyertakan nilai HEX, RGB, HSL, persentase *coverage*, dan nilai *confidence* dari model. Format CSS Variables dan konfigurasi Tailwind CSS dihasilkan sebagai format pendukung yang dapat langsung digunakan di proyek *front-end* tanpa modifikasi tambahan.


```
{
  "color": {
    "background": { "value": "#fbfbfb", "coverage": "84.6%", "confidence": 0.93 },
    "accent": { "value": "#92c14e", "coverage": "5.9%", "confidence": 1.00 },
    "border": { "value": "#c7c9c5", "coverage": "2.6%", "confidence": 1.00 },
    "primary": { "value": "#feb867", "coverage": "2.4%", "confidence": 1.00 },
    "text": { "value": "#202723", "coverage": "2.4%", "confidence": 1.00 },
    "secondary": { "value": "#5d7377", "coverage": "2.1%", "confidence": 1.00 }
  },
  "$metadata": {
    "source_image": "test01", "theme": "light", "total_colors": 6
  }
}
```


Gambar 3. Potongan Output JSON Token

Selain JSON, sistem juga menghasilkan CSS Variables dalam format :root { --color-primary: #feb867; } dan konfigurasi Tailwind CSS dalam format module.exports, keduanya tersimpan otomatis di folder output/ dan siap digunakan langsung di proyek *front-end* berbasis.

F. User Acceptance testing (UAT)

Validasi sistem dilakukan melalui UAT terhadap 5 responden yang memiliki pengalaman di bidang desain UI/UX, pengembangan *front-end*, atau keduanya. Setiap responden diminta mencoba sistem secara langsung dengan mengunggah gambar antarmuka pilihan mereka, lalu mengisi kuesioner penilaian menggunakan

skala Likert 1–5. Pengujian mencakup lima aspek yang ditampilkan pada Tabel 6.

Tabel 6. *Profil Responden UAT*

Responden	Latar belakang
R1	Desainer UI/UX
R2	Developer Front-End
R3	Desainer UI/UX & Developer Front-End
R4	Desainer UI/UX
R5	Developer Front-End

Tabel 7. *Hasil Penilaian UAT per Responden (Skala Likert 1–5)*

Aspek Penilaian	R1	R2	R3	R4	R5	Avg
Relevansi Warna	5	4	4	5	4	4.40
Kesesuaian warna	5	5	5	4	5	4.80
Kegunaan	4	4	4	4	5	4.20
Kelengkapan Informasi	4	5	4	5	4	4.40
Kepercayaan Hasil	5	5	5	4	5	4.80
Weight Avg	4.60	4.60	4.40	4.40	4.60	**4.52**

Hasil UAT menunjukkan rata-rata skor keseluruhan sebesar 4,52 dari 5. Dua aspek mendapat skor tertinggi yaitu kesesuaian peran warna dan kepercayaan hasil (masing-masing 4,80), menunjukkan bahwa klasifikasi peran yang dihasilkan dinilai akurat dan layak dijadikan acuan dalam proyek nyata. Aspek kegunaan format output mendapat skor terendah (4,20), yang dapat dipahami karena responden menilai output tanpa mencoba mengintegrasikannya langsung ke proyek. Secara keseluruhan, sistem diterima dengan baik oleh seluruh responden..

IV. KESIMPULAN

Sistem *Color Guide Generator* berhasil dikembangkan menggunakan kombinasi K-Means *Clustering* untuk dominan dan *Decision Tree* untuk klasifikasi peran desain, dengan output berupa *design token* dalam format JSON, CSS *Variables*, dan konfigurasi Tailwind CSS. Model mencapai F1-*score* 0,98 untuk mode terang dan 0,92 untuk mode gelap, dengan *Silhouette Score* K-Means sebesar 0,8656, keduanya melampaui target kelayakan yang ditetapkan.

Validasi UAT terhadap 5 responden menghasilkan rata-rata skor 4,52 dari 5, menunjukkan sistem diterima baik dan mampu menggantikan proses penentuan warna manual secara efektif. Pengembangan selanjutnya dapat diarahkan pada perluasan dataset mode gelap yang lebih akurat dan dukungan ekspor ke format Figma tokens atau Flutter.

UCAPAN TERIMA KASIH

Kepada Universitas Duta Bangsa yang telah memberikan kesempatan sehingga pembuatan jurnal ini bisa terlaksana. Selanjutnya kepada peneliti terdahulu yang telah menjadi referensi bagi penulis untuk melakukan penelitian ini.

REFERENSI

[1] S. A. R. Refiana, A. G. Tommy, and H. Kremer, "Analisis Warna pada Desain Situs Web Sociolla & Guardian," Jurnal Rupa Matra, vol. 3, no. 2, pp. 145–157, 2025, doi: 10.62375/jdkv.v3i2.509.

[2] Z. Z. Maulidia and R. Andrian, "Perancangan Website Majalengka Saber Hoaks dalam Mendukung Proses Verifikasi Informasi dengan Menggunakan Metode Design Thinking," Jurnal Teknologi dan Informasi, vol. 13, no. 1, 2023, doi: 10.34010/jati.v13i1.

[3] W. Wirdawati, S. Yulihartati, and A. Ramadhanu, "Implementasi Euclidean Distance dan Segmentasi K-Means Clustering pada Identifikasi Citra Jeruk Nipis," Indonesian Journal Computer Science, vol. 3, no. 2, pp. 72–79, 2024, doi: 10.31294/ijcs.v3i2.5600.

[4] Fauziah, Y. Elviralita, W. Wisanty, and Isminarti, "Prosiding Seminar Nasional Teknik Elektro dan Informatika (SNTEI) 2023 – Teknik Informatika," 2023.

[5] S. Arifah, E. R. Swedia, and M. R. D. Septian, "Analisis Perbandingan Algoritma Clustering dalam Melakukan Segmentasi Warna pada Citra Jajan Tradisional," Sebatik, vol. 27, no. 1, pp. 70–76, 2023, doi: 10.46984/sebatik.v27i1.2273.

[6] A. Ma'ruf, N. Mardiyantoro, and H. Sibyan, "Ekstraksi Palet Warna untuk Kompresi Gambar Digital menggunakan Algoritma K-Means," 2025.

[7] L. Riani et al., "SiCitra: Aplikasi Berbasis Web untuk Pemrosesan Citra Digital Menggunakan OpenCV," Jurnal Informatika Upgris, vol. 10, no. 2, pp. 39–46, 2024, doi: 10.26877/jiu.v10i2.20924.

[8] M. I. Manzis and R. P. Aji, "Implementasi Machine Learning Berbasis Fitur Warna RGB dan HSV untuk Klasifikasi Kualitas Air," Jurnal Informatika dan Teknik Elektro Terapan, vol. 14, no. 1, 2026, doi: 10.23960/jitet.v14i1.8667.

[9] Agita Nurfadillah, R. Andarsyah, and R. M. Awangga, "Sistem Rekomendasi Warna Kontekstual untuk Desain UI/UX Menggunakan Random Forest," Jurnal Teknologi dan Manajemen Industri Terapan, vol. 4, no. 3, pp. 777–783, 2025, doi: 10.55826/jtmit.v4i3.1023.

[10] A. R. Setiawan, M. Asfi, A. Sevtiana, S. Pranata, and W. E. Septian, "Design System pada Perancangan Antarmuka Perangkat Lunak Sistem Akses Digital," Jurnal Teknologi Terpadu, vol. 9, no. 1, pp. 56–64, 2023, doi: 10.54914/jtt.v9i1.619.

[11] D. L. Amrullah, E. R. Swedia, M. Cahyanti, and M. R. D. Septian, "Implementasi Color Detection Menggunakan

Algoritma Midpoint Berbasis Sistem Operasi Android," Sebatik, vol. 26, no. 1, pp. 121–130, 2022, doi: 10.46984/sebatik.v26i1.1631.

[12] A. Maolana, "Analisis Perbandingan Hasil Kompresi Citra Format PNG dengan SVG untuk Penyimpanan File Gambar," Dinamik, vol. 29, no. 1, 2024, doi: 10.35315/dinamik.v29i1.9362.

[13] T. Rahmawati, Y. Wilandari, and P. Kartikasari, "Analisis Perbandingan Silhouette Coefficient dan Metode Elbow pada Pengelompokkan Provinsi di Indonesia Berdasarkan Indikator IPM dengan K-Medoids," Jurnal Gaussian, vol. 13, no. 1, pp. 13–24, 2024, doi: 10.14710/j.gauss.13.1.13-24.

[14] N. Yasmin, S. C. D. Akbar, and A. Ramadhanu, "Penerapan K-Means Clustering untuk Klasifikasi Citra Cabai Keriting: Studi Ekstraksi Warna dan Tekstur GLCM," Indonesian Journal Computer Science, vol. 3, no. 2, pp. 65–71, 2024, doi: 10.31294/ijcs.v3i2.5758.

[15] S. Saputra, "Analisis Perbandingan Format Pesan ISO 8583 dan JSON pada Sistem Transaksi Keuangan," Bulletin of Computer Science Research, vol. 6, no. 1, pp. 548–557, 2025, doi: 10.47065/bulletincsr.v6i1.839.