ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

Color Guard: Automating Google Material Design Color Guidelines Compliance for Mobile Applications (January 2025)

Sajani Anupama Balasooriya[1], Jagath Wickramarathne[2]

[1]Faculty of Graduate Studies, [2]Faculty of Computing,
[1,2]Sri Lanka Institute of Information Technology, Malabe, Sri Lanka

Publication Date: 2025/04/05

Abstract: Developing mobile applications today demands significant time and effort. Creating user-friendly user interfaces (UIs) is particularly challenging, with special attention needed for color-related details as they are the first thing customers notice. Numerous guidelines have been introduced to assist mobile UI designers in fostering good interaction between users and UIs. Among these, Google's Material Design guidelines are highly trusted, developed, and maintained by Google. Adhering to these guidelines enables developers and designers to create more efficient and effective UIs, which is crucial for commercial mobile applications. However, reading, understanding, and implementing all these guidelines can overwhelm novice UI designers. Additionally, having improvement tips and suggestions is highly beneficial. To address this challenge, this research proposes a web-based solution that reviews developed mobile UIs and provides textual suggestions to improve the UI design according to the design guidelines to support novice designers in designing their UIs more user-friendly. A solution is implemented as a web application, offering an effective way to provide this service across any device and operating system.

Keywords: *Mobile Applications, Design Guidelines, UI Element Colors, UX.*

How to Cite: Sajani Anupama Balasooriya, Jagath Wickramarathne. (2025). Color Guard: Automating Google Material Design Color Guidelines Compliance for Mobile Applications (January 2025). *International Journal of Innovative Science and Research Technology*, 10(3), 2185-2202. https://doi.org/10.38124/ijisrt/25mar1435.

I. INTRODUCTION

In today's digital world, mobile applications have become essential for both consumers and businesses. They shape our daily activities, streamline processes, and keep us connected. With the rise of smartphones and mobile devices, there's a growing demand for apps that offer convenience, efficiency, and engagement. We rely on these apps for everything from checking the weather and managing finances to staying in touch with loved ones and shopping online. Mobile apps are powerful for businesses to reach more people, build customer loyalty, and provide personalized experiences that enhance their brand.

The increasing importance of mobile apps reflects their role in our everyday lives. They have changed how we interact with technology, offering solutions that are always within reach and tailored to our needs. Mobile apps provide various functions, whether social media, online shopping, health tracking, or learning. This makes them a keyway for businesses to engage effectively with their customers. Moreover, apps help companies to run more smoothly, improve customer service, and use data to make better decisions. The design and functionality of these apps are crucial for standing out in a crowded market.

User interfaces (UIs) are at the heart of the user experience (UX) and satisfaction with mobile apps. A well-designed UI is intuitive, visually appealing, and easy to navigate, allowing users to accomplish their tasks effortlessly. It is the bridge between the user and the app, and its design affects how users feel about the app. A messy or complicated UI can frustrate users, while a clean and user-friendly interface can make them more engaged and satisfied. Key aspects of good UI design include responsive design, consistent layout, straightforward typography, and the effective use of colors and icons, all working together to create a seamless experience.

When we implement an automated system, we need to focus not only on completing the task but also on the usability of the system as well. The users interact with the system through the UI, and for users, the UI is the system. That is why we evaluate the user experience with the system, check where the difficulties are, and find ways to improve it. At this point, the UI design guidelines come to the scene with

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

properly formatted, user-friendly regulations to follow for better UI designs.

In the competitive world of mobile apps, the quality of the UI can make or break an app's success. Apps focusing on user-friendly design are more likely to retain users, receive positive reviews, and achieve higher conversion rates. Additionally, considering accessibility in UI design ensures that apps are usable by people with different abilities, making them more inclusive. Investing in high-quality UI design isn't just about making an app look good—it's about creating a functional and enjoyable experience that meets users' needs and expectations, ultimately leading to greater satisfaction and loyalty.

Mobile application development takes time and much more effort. When it comes to developing user-friendly UIs, it is more complex than creating the backend and integrating with the frontend and backend of the code. Visually appealing and well-coordinated UI enhances user experience, fosters brand identity, and ultimately contributes to the success of a mobile application. Currently, several guidelines have been introduced for mobile UI designers and developers to help them create effective Human-Computer Interactions between users and mobile interfaces. For example, Google's material design, Android's design guidelines, Apple's Human Interface Guidelines and Microsoft's Fluent Design System can be taken. By following those guidelines, the developers and designers can make more efficient and effective UIs. Even if violating these guidelines does not cause bugs in the mobile application, it can significantly impact usability and user experience [1]. Therefore, adhering to these guidelines is crucial when developing commercial mobile application UIs. However, reading, understanding, and following all these guidelines are challenging for novice designers.

However, creating a flawless UI often faces challenges in maintaining consistency across different application elements, especially concerning Color schemes and Themes. Inconsistencies in colors and themes can occur due to various reasons, such as neglected details, human errors during implementation, or evolving design requirements over time. These discrepancies can negatively impact the application's overall aesthetic appeal, usability, and brand coherence. Manual detection and resolution of such issues are time-consuming, error-prone, and often require substantial design expertise.

To address this challenge, we propose a web-based solution called ColorGuard, designed to review developed mobile UIs and provide textual suggestions for enhancing UI design by established guidelines. This solution will be implemented as a web application, offering an efficient method to deliver services across any device and operating system.

In recent years, machine learning (ML) techniques have opened new avenues for automating various UI design and development aspects. By harnessing the capabilities of ML models, the proposed system aims to identify inconsistencies in color schemes and themes across different UI elements within a mobile application and then analyze the context and significance of detected issues based on design principles and user preferences. Then, using that, generate suggestions to harmonize color schemes and themes, ensuring visual coherence and enhancing user experience, and provide insights and feedback to designers for continuous improvement in UI design practices.

Google Material Design Guidelines [2] play a crucial role in modern app design, offering a comprehensive framework that helps developers create cohesive and visually appealing user experiences. Introduced by Google, Material Design is a design language that combines classic principles of good design with the innovation and possibility of technology and science. It aims to create a unified experience across all platforms and device sizes, providing a consistent and intuitive user interface for devices with Android operating systems.

One of the standout features of Material Design is its emphasis on using color and theme guidelines effectively. Color is not just a decorative element but a functional one that guides user interactions and enhances usability [3]. Material Design provides a detailed color palette that helps developers and designers choose colors that work well together, ensuring visual harmony and accessibility. These guidelines also emphasize the importance of contrast and readability, which are critical for creating interfaces that are easy to navigate and understand.

Adhering to design guidelines, such as Google's Material Design, poses several significant challenges. One considerable area for improvement is understanding and correctly applying these guidelines. Design guidelines are often comprehensive and detailed, which can be overwhelming for developers, especially those new to them. The specifics of these guidelines require careful study and precise implementation. Misunderstanding or partial understanding can lead to inconsistencies or flawed designs that fail to adhere to the intended standards, ultimately compromising the user experience.

Moreover, the process of manually checking compliance is inherently time-consuming and prone to errors. Ensuring each aspect of an app meets the specified guidelines involves meticulous scrutiny and repeated verification. This manual process is labor-intensive, diverting valuable time and resources from other critical development tasks. Additionally, the human element introduces a risk of oversight and mistakes. Even experienced developers can miss the important details or incorrectly apply guidelines, leading to non-compliance and necessitating further corrections. These corrections extend the development timeline and increase costs.

These challenges underscore the need for more efficient methods and tools to support guideline compliance. Automating parts of the compliance process checking can significantly mitigate these issues, offering faster, more accurate, and reliable adherence to design standards.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

However, achieving this requires a balance between leveraging automated solutions and maintaining a thorough understanding of the guidelines to ensure that the automated tools are applied effectively.

II. RELATED WORK

Compliance with mobile UI guidelines helps develop user-friendly and accessible applications. Traditional approaches include manual audits and checklists, which are time-consuming and prone to errors. In the current context, researchers are considering advanced technologies such as AI, machine learning, and computer vision to make the process easier. This research focuses on using tools such as Generative Adversarial Networks (GANs) and Convolutional Neural Networks (CNNs) to detect UI issues, generate design recommendations, and enhance user experiences, thus offering efficient and scalable solutions for modern UI design.

A. Guidelines Checking

Several researches have been conducted to identify UI components, detect UI issues, and generate new UI designs. Most of these researches give the output as UI issues report or violated guidelines report [4], [5]. When a UI designer or a developer works on a UI, it is essential to understand where the UI issues are and what the UI issues are. So, these existing researchers have done a great job covering this requirement.

Extensive research has been conducted to detect UI issues and ensure adherence to design guidelines. Among these efforts, the UIs Hunter web application stands out for its significant contribution [4]. This research involved developing and training a proprietary model capable of detecting UI issues. When users upload a UI for evaluation, the proposed application analyzes it and generates a comprehensive issue report. The specialty is that the user can select the guideline categories like component, design aspect, component knowledge aspect, and severity. According to the chosen guideline category in Google Material Design Guidelines, the web application will give the user a list of violated guidelines as feedback.

The subsequent eye-catching research is called "Owl Eyes: Spotting UI Display Issues via Visual Understanding" [5], developed for UI testing. To identify UIs with issues, they used the CNN (Convolutional Neural Network) model and Grad-CAM (Gradient weight Class Activation Mapping) to localize the areas with UI issues. The specialty is that researchers of the Owl Eyes project have focused on four categories of UI issues, which are component conclusion, text overlap, missing images, null values, and blurred screen. This project's input is mobile screenshots, and the output will be presented as a text-based issues report.

The research on "Wireframe-based UI Design Search through Image Autoencoder" [6] is innovative as it involves training a wireframe image autoencoder using an extensive database of real-application UI designs. The relevant UI designs do not need to be labeled. The researchers have implemented this approach for Android UI design search and conducted extensive experiments using artificially created relevant UI designs. They also conducted human evaluation of UI design search results.

Ensuring guideline compliance in the ColorGuard app typically involves audits, checklists, training, and peer reviews. Regular audits are conducted periodically to review workflows and procedures, ensuring they align with the guidelines. This method, while thorough, is time-consuming and requires a substantial workforce. Checklists ensure that every step in the process adheres to the guidelines, but these can become lengthy and cumbersome. Under time pressure, there is a significant risk of items needing to be noticed. Training sessions and workshops are essential to keep staff updated on design guidelines [7]. Still, the effectiveness of this method relies heavily on staff retention and application of knowledge. These sessions can also be costly and require a significant time investment. Peer reviews involve employees reviewing each other's work to ensure compliance, but this method can lead to conflicts and may only sometimes be reliable [8]. Additionally, maintaining manual records is a common practice. Still, it is prone to human error, with risks of records needing to be completed or recovered, making this method inefficient and time-consuming [9].

There is an existing research called "GUIGAN: Learning to Generate GUI Designs Using Generative Adversarial Networks" [10], which is a method for generating GUI designs based on Generative Adversarial Networks (GAN). This research aims to automatically create UI designs by selecting a list of existing GUI component subtrees to compose new GUI designs for designers and developers. Users can upload their mobile UI screenshots, and after analysis, they can receive designed images. These generated images can be taken as a starting point or inspiration for their design work. This model can develop other parts of the GUI to complete the whole GUI design, given some pre-built GUI components, such as navigation and pictures. The generated samples have different and diverse styles and structures.

B. Limitations in Manual Methods

Manual methods for ensuring compliance come with several limitations. The most significant limitation is human error [9]. Mistakes can easily be made, especially when staff are under pressure or if there is an oversight, leading to non-compliance with guidelines. Manual processes are often slow and labor-intensive, resulting in inefficiency and delays. Ensuring consistency across individuals and teams can be challenging, leading to variable compliance levels. As organizations grow, manual methods become increasingly impractical and more challenging to manage, posing scalability issues. Manual record-keeping is particularly prone to inaccuracies and can be challenging to organize and retrieve, further complicating the compliance process. These limitations underscore the need for more efficient and reliable methods to ensure guideline compliance.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

C. Benefits of Automated Tools

Automated tools offer significant advantages over manual methods in ensuring guideline compliance, particularly in the ColorGuard application. One of the primary benefits is speed. Automated tools can perform tasks much faster than humans, ensuring guidelines are followed in real time [11], [12]. This reduces the likelihood of delays and increases overall efficiency. Accuracy is another critical benefit. Automated systems significantly reduce the chances of human error, providing more reliable compliance with guidelines [13]. Consistency is also improved as automated systems apply the same rules uniformly, ensuring that all processes adhere to the guidelines consistently [14].

Efficiency is markedly enhanced with automated tools, as they free up human resources to focus on more strategic tasks, reducing the overall workload. These tools are also highly scalable and can manage increased workloads as the organization grows without requiring additional human resources. Automated tools provide real-time monitoring and alerts, allowing immediate corrective actions if guidelines are not followed. This capability is crucial in maintaining continuous compliance and preventing issues before they escalate. Comprehensive reporting is another advantage, as automated tools can generate detailed reports and analytics, providing insights into compliance levels and identifying areas needing improvement. While there is an initial investment in automated tools, they are cost-effective in the long term, reducing the costs associated with manual compliance methods and enhancing overall productivity and reliability [15].

Jeff Hawkins was the first person to develop a handheld device idea [16]. He used to bring a wooden item everywhere he went, pretending to be a hand-held device, and he pretended to do everything using that. His imagination was the key to making the current mobile devices.

D. Importance of Focusing on Mobile UIS

Today, many mobile applications are available for different industries, handling tasks once done in person, like grocery shopping and banking. Most of these apps are used by non-IT professionals and older people. So, having user-friendly UIs that follow human-computer interaction (HCI) concepts and have an understandable process flow is essential [17]. When it comes to user-friendly interfaces, UI designers and frontend developers are responsible for designing and developing understandable user interfaces.

E. Different Ui Design Guidelines

To design and develop user-friendly UIs, several UI design guidelines have been introduced and explained by different people from different perspectives [18]. Following mobile UI design guidelines ultimately results in better accessibility, user satisfaction, and overall success of mobile applications. There are very commonly used UI design guidelines for mobile applications like Google Material Design Guidelines [19], Android Developers Design Guidelines [2], Apple's Human Interface Guidelines [20], [21] and Microsoft's Fluent Design System [22] are available.

The paper by Awwad et al. (2017) presents a study focused on enhancing the usability of the Pocket Paint application by aligning it with Material Design principles and incorporating support for internationalization and localization at the application level [23]. The authors of [23] detail the process of redesigning the user interface to adhere to Material Design guidelines, aiming to create a more intuitive and visually appealing experience. Additionally, they emphasize the importance of making the app accessible to a global audience by integrating features that support multiple languages and regional preferences. The study demonstrates that these improvements improve user satisfaction and usability, highlighting the significance of design compliance and localization in mobile application development.

Another research has been conducted by Kalac, Borovina, and Boskovic (2021) delves into the challenge of balancing traditional interaction design principles with the sleek, modern guidelines of Material Design [24]. The paper highlights how designers can effectively blend the best of both worlds. They provide insights and examples on maintaining usability and a great user experience while adopting Material Design's structured, visually appealing approach. The study underscores the importance of not sacrificing user-friendly interactions for the sake of trendy design, aiming to help designers create beautiful and easy applications.

Ensuring compliance with mobile UI design guidelines is crucial in delivering accessible and user-friendly applications. While traditional methods have their merits, they often need to improve efficiency and scalability. By integrating advanced technologies like AI, machine learning, and computer vision, this research offers innovative solutions to streamline compliance processes. Tools such as GANs and CNNs provide the ability to detect UI issues and generate design recommendations with greater accuracy and consistency. These advancements pave the way for more efficient, scalable, and user-centric approaches to mobile UI design, ultimately enhancing user satisfaction and application success.

F. Google Material Design Guidelines

Despite the availability of these different guidelines, we often choose to go with Google's Material Design guidelines for several reasons. Firstly, Material Design provides a robust and comprehensive framework that covers a wide range of design elements [25], from essential components to complex interactions [26]. Its detailed documentation and resources make it accessible to both novice and experienced designers. Android is also the most commonly used operating system for mobile devices [27]. Additionally, Material Design's principles are based on extensive research and usability testing, ensuring that the guidelines are grounded in best practices for user experience [28].

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

Secondly, Material Design offers a high degree of flexibility and adaptability, allowing designers to customize and extend the guidelines to fit their specific needs [29]. This flexibility makes it suitable for various applications [30], from simple mobile apps to complex web interfaces. Material Design's emphasis on a unified experience across different platforms ensures that applications designed with these guidelines are consistent and coherent, regardless of the device used.

Lastly, Google's Material Design has a strong community and ecosystem support [31]. With a vast array of tools, libraries, and frameworks, designers and developers can quickly implement Material Design principles in their projects. The continuous updates and improvements from Google also ensure that the guidelines remain relevant and up-to-date with the latest design trends and technological advancements.

G. Google Material Design – Color Guidelines

Google's Material Design Guidelines, several guidelines have been defined regarding mobile UI color and themes [19]. We have decided to move forward with the color and theme category out of all eight available categories because colors and themes are the most immediately noticeable user interface elements. They play a crucial role in creating the first impression of an application, influencing user engagement and satisfaction. Under color and theme guidelines, eight guidelines are defined. Those guidelines are determined according to the UI element, as well as the primary and secondary colors.

In UI/UX design, primary colors represent the main brand identity and are used prominently in key elements like buttons, icons, and highlights to draw attention. Secondary colors complement the primary palette, providing contrast and variety for backgrounds, text, or secondary actions, ensuring visual hierarchy and usability.

➢ *The Guidelines are as Follows:*

- In mobile applications, the top and bottom bars, known as app bars, are designed using the app's main color to maintain a consistent and recognizable look. To help distinguish between the app content and the device's system interface, the system bars (which show the time, battery, and other system indicators) can be shaded in either a darker or lighter version of the app's primary color. This creates a clear visual separation. Additionally, to make the app bars stand out more distinctly from other elements on the screen, nearby components like the floating action button (FAB) can be colored using a secondary color, providing a contrasting and visually appealing design (Figure 1).

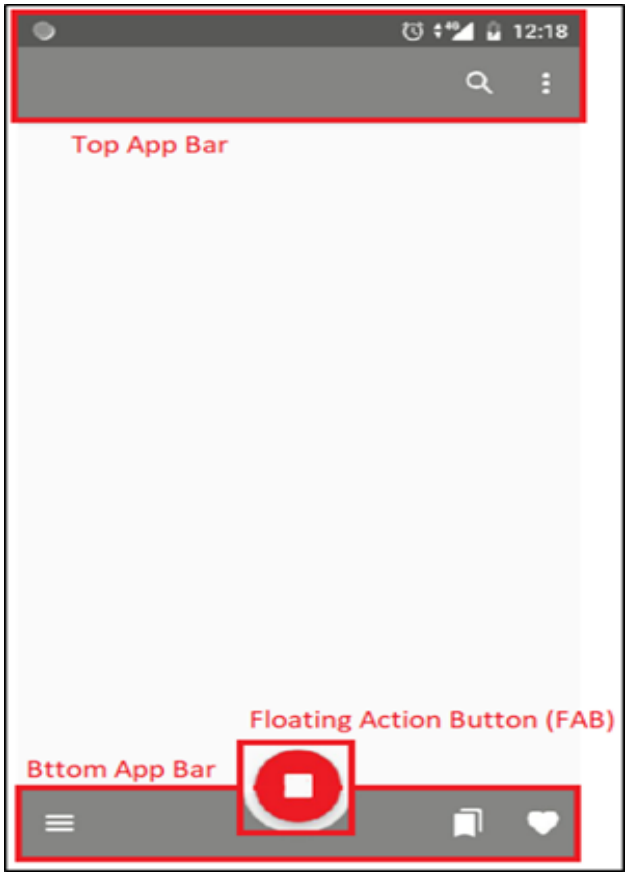

Fig 1: Example for Top app bar, Bottom app bar, and FAB

- When the color of an app's top or bottom bar matches the background color, these bars visually merge with the background. This blending effect minimizes the prominence of the app's structural elements, allowing the app's content to take center stage. By creating a seamless transition between the app bar and the background, the design directly draws the user's attention to the content, enhancing engagement and providing a more immersive and fluid user experience. This approach can make the app feel more cohesive and less cluttered, emphasizing the importance of the app's content over its navigational or structural elements.
- In an app's UI, there are typically two layers: the front and back layers (Figure 2). To clearly differentiate between these layers, the back layer is usually colored in the app's primary color, while the front layer is set to white. This color scheme helps users quickly identify and distinguish between the different layers, enhancing the overall clarity and usability of the app. The primary-colored back layer provides a strong foundation and visual depth, while the white front layer ensures that content is easily readable and stands out against the background.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

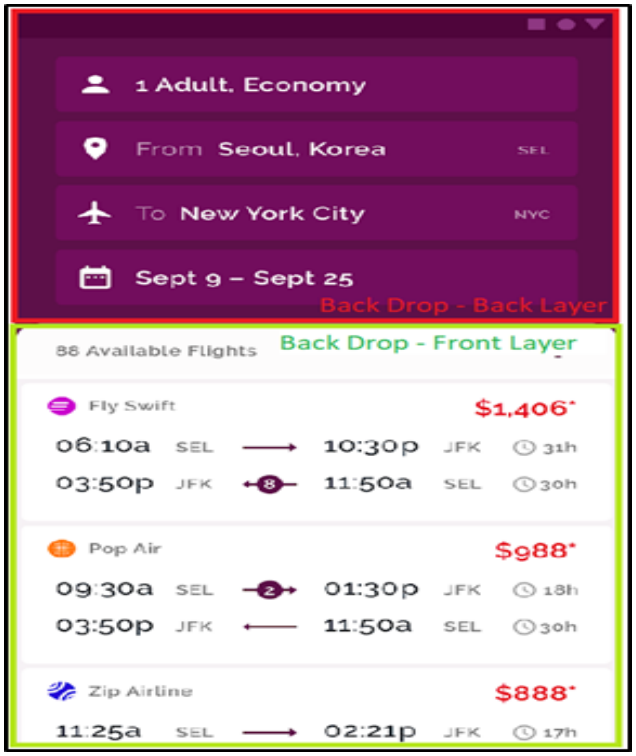

Fig 2: Example for Backdrop

- In an app's user interface, sheets and surface elements such as bottom sheets, navigation drawers, menus, dialogs, and cards should use white as their baseline color (figure 3). This choice of color ensures that these UI elements are clean and easy to read, providing a neutral background that allows content to stand out clearly. Using white for these components creates a consistent and visually appealing look across the app, enhancing usability and ensuring that these interactive elements are easily distinguishable from other parts of the interface.

Fig 3: Example for Sheets and Surface Elements

- Modal sheets (Figure 4), such as navigation drawers and bottom sheets, which appear temporarily on the screen, should use contrasting colors to stand out. While these surfaces are typically white, you can use your app's primary or secondary color to create a distinctive and visually appealing contrast. This approach ensures that modal sheets are easily noticeable and visually separate from the underlying content, enhancing user interaction and overall app experience.

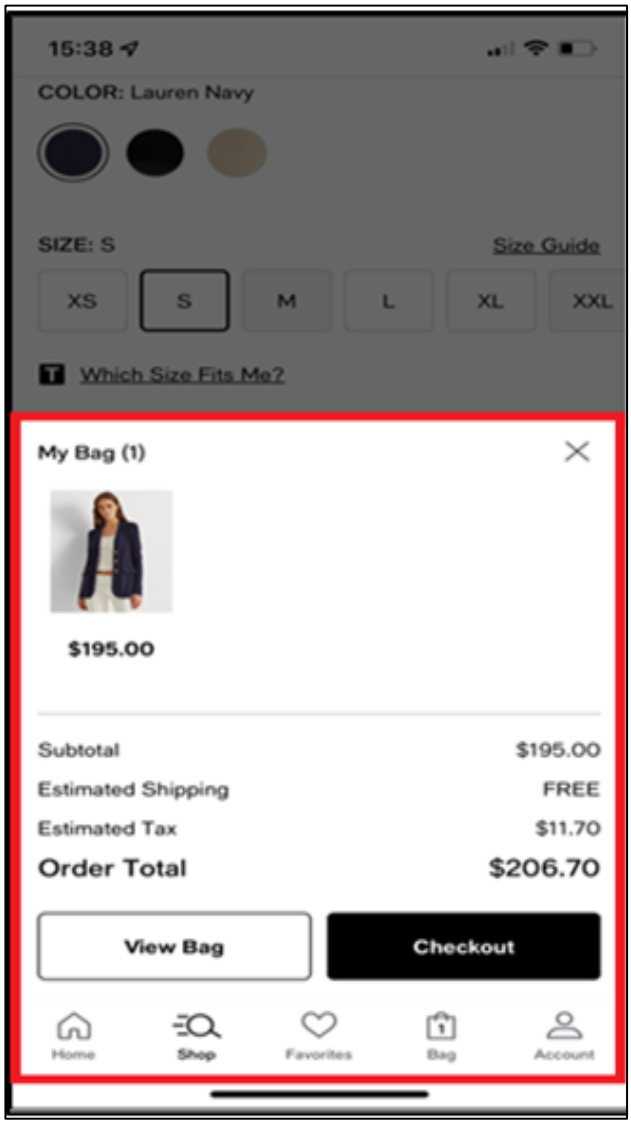

Fig 4: Example for Modal Sheets

- The baseline color for cards in an app's interface is white. However, this color can be customized to reflect the brand's identity better or enhance legibility. Additionally, the text and icons on the cards can be adapted to match the app's color theme, further improving readability and ensuring a cohesive look throughout the app. This flexibility allows consistent and brand-specific design, improving the overall user experience.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

- In an app's user interface, the baseline color for contained, text, and outlined buttons is the app's primary color, ensuring these buttons stand out and are easily identifiable. Floating action buttons (FABs) and extended floating action buttons (Extended FABs) use the app's secondary color, making them distinct and visually appealing. Similarly, selection controls, such as checkboxes, radio buttons, and switches, also use the secondary color, providing a consistent and cohesive look throughout the app. These color guidelines help maintain a visually appealing and user-friendly interface, enhancing overall usability and aesthetic coherence.

This chapter has reviewed how leading guidelines for UI design shape digital interfaces across various platforms. Each brings unique philosophies and frameworks to the attention of modern usability needs. Material Design by Google, HIG (Human Interface Guidelines) by Apple, and Fluent Design System by Microsoft have improved beauty expectations through a continuing evolution of functional design principles promoting accessibility, consistency, and user-centered interactions.

These guidelines hint at principles that assure intuitive and engaging user experiences. Material design offers both adaptability and community and inspires a powerful kit upon which to build cohesively across devices to ensure the scale balance between flexibility and usability. HIG underlines Apple's commitment to simplicity and performance within its ecosystem and shows how users' interests can be upgraded with minimal presentation. Through its layered and adaptable components, Fluent Design is how Microsoft has been able to establish its commitment to inclusivity and accessibility, thereby opening up the way to immersive and functional interfaces across device types.

Ultimately, all these design frameworks epitomize how deliberate design choices can redefine the digital experience. From color theory to component layering, each of these guidelines offers a different perspective on designing appealingly beautiful and functional UIs. In the future, established frameworks will continue to be influenced by the incoming trends in technology, and evolved user expectations are instrumental in setting standards for tomorrow's digital interactions. Knowledge applied to understand and apply these design systems forms a foundation for designers and developers to create impactful, user-friendly, and aesthetically appealing digital UIs.

III. RESEARCH OBJECTIVES

This research aims to develop a web-based system that identifies UI design issues and provides text-based improvement suggestions based on Google Material Design color and theme guidelines. This system aims to automate the identification of design inconsistencies, significantly reducing the manual effort typically required to ensure adherence to these guidelines. Doing so not only streamlines the design process but also empowers novice designers to enhance their work by providing concrete, guideline-based feedback.

One of the key sub-objectives of this research is to automate the detection of UI design issues according to established guidelines, thereby minimizing the manual effort involved in reviewing and correcting these issues. This automation will help in quickly identifying areas of non-compliance and suggesting improvements, making the design process more efficient and less prone to human error. The automated UI design validation system will also enhance the overall user experience by enabling even novice designers to create high-quality, compliant designs.

Furthermore, as a web-based application, this system is designed to be accessible globally once it is hosted. This cross-platform functionality ensures that designers from diverse backgrounds and with different technical setups can benefit from the tool, making it a versatile resource for improving UI design standards universally.

By addressing the above-mentioned objectives, the research aims to contribute significantly to the UI/UX design field by promoting the creation of visually appealing and user-friendly interfaces that adhere to established guidelines. The system's capability to provide immediate, text-based suggestions based on Google Material Design principles will foster a more intuitive and efficient design process, ultimately leading to higher-quality digital products.

IV. RESEARCH METHODOLOGY

In this section, we are going to discuss the steps that we have followed in order to develop the application. Firstly, we describe the system diagram we followed and then explain the datasets we used to create the model. After that, we mentioned the steps we followed to develop the backend: model training, including the tools and technologies we used. After that, we mentioned the frontend development steps, as well as the tools and technologies. Finally, we have mentioned the steps for usage and how a novice designer can use this system.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

A. System Diagram

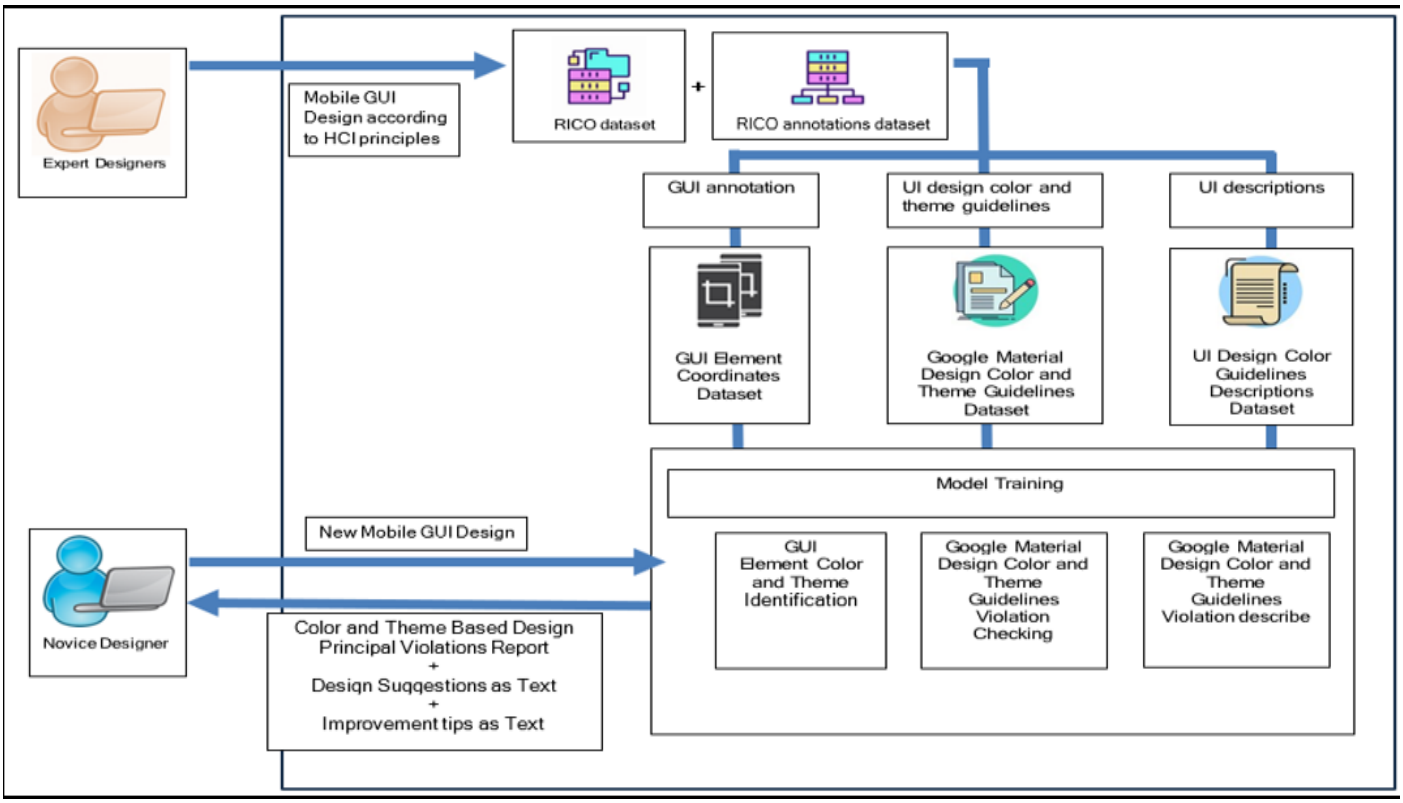

Fig 5: Proposed System Diagram

Figure 5 illustrates a comprehensive and user-friendly process designed to evaluate and improve mobile UI designs, focusing on adhering to Google Material Design Guidelines for color and theme.

At the heart of this system are two key user groups: Expert Designers and Novice Designers. Expert Designers contribute by creating mobile GUIs that strictly follow HCI principles. These expertly crafted designs serve as a valuable reference point for training the system, ensuring that the model learns from the best practices in the industry.

The system relies on several essential datasets to function effectively. The RICO Dataset [8], a vast collection of mobile UI screenshots, provides a rich visual data source for analysis. The RICO Annotation Dataset offers detailed annotations of the UI elements within the RICO Dataset, including their specific coordinates and other relevant details [9]. This information is crucial for understanding the structure and placement of UI components.

Moreover, the system integrates the UI Design Color and Theme Guidelines, primarily sourced from Google's Material Design. These guidelines ensure that the system's evaluations are aligned with widely accepted industry standards, promoting consistency and proficiency in design. Additionally, the UI Descriptions dataset provides textual descriptions and details about various UI elements, adding a layer of contextual understanding to the visual data.

The model training process is a sophisticated integration of these datasets. It begins with GUI annotations, focusing on the precise coordinates of UI elements to grasp their layout and structure. The Color and Theme Guidelines dataset teaches the model to recognize and adhere to the specific color schemes and thematic guidelines set by Google Material Design. The UI Descriptions dataset further enhances the model's understanding by providing context about the purpose and correct implementation of different UI components.

Once trained, the system is ready to assist Novice Designers. When a novice designer submits a new mobile GUI design, the system analyzes it, checking for compliance with the established color and theme guidelines. If any violations are detected, the system generates a detailed report highlighting these issues. Furthermore, it offers practical design suggestions and textual improvement tips, guiding novice designers in refining their UI to meet professional standards.

This system combines the expertise of experienced designers, extensive datasets, and advanced machine-learning techniques to assist novice designers. By delivering real-time feedback and practical suggestions, it enables them to create visually appealing and guideline-compliant mobile UI designs, raising the overall quality standard in the industry.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

B. Used Datasets

There are two datasets that has been used to implement the web application and those datasets are as follows.

The RICO dataset [32] is used for model training. It contains more than 27 categories of UI screenshots, exposing the visual, textual, structural, and interactive design properties of more than 66,000 unique UI screenshots.

An annotated dataset was also used for RICO data screenshots [33]. It consists of pairs of mobile screenshots and their annotations. The annotations are in text format and contain information on the UI elements present on the screen: their type, their location, the text they contain, or a short description. The training dataset has more than 15000 data. As this RICO annotations dataset did not cover several required elements for this project (e.g., top app bar, bottom app bar, etc.), we modified it with a simple Python code to make a complete dataset. The dataset previews are shown below in Figure 6.

	A	B	C	D	E	F	G	H
1	screen_id	screen_annotation						
2	100.jpg	(406, 912, 438, 946, "Button"), (253, 911, 288, 947, "Button"), (102, 910, 134,						
3	10008.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828,						
4	1001.jpg	(406, 912, 438, 946, "Button"), (253, 911, 288, 947, "Button"), (102, 910, 134, 9						
5	10015.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 182						
6	10038.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828,						
7	10041.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828						
8	10045.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 182						
9	10047.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828,						
10	10050.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828						
11	10053.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 182						
12	10054.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828,						
13	1006.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828						
14	10062.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 182						
15	10064.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828,						
16	10066.jpg	(817, 1830, 872, 1885, "Button"), (511, 1828, 570, 1887, "Button"), (210, 1828						

Fig 6: RICO Annotations Dataset – Part of the Dataset

The RICO dataset contains no annotations, specifically the UI element locations. However, for model training purposes, the exact locations of the UI elements need to be known. That is why we needed this second dataset containing the RICO dataset's UI annotations.

C. Model Training

Google Colab is the platform used to train the model as it performs better than Jupyter Notebook. TensorFlow has been used to analyze and process images to verify compliance with design guidelines. TensorFlow's `tf.keras.utils.Sequence` is the class used to create data generators for training models on large datasets and help handle large amounts of image data efficiently during training. The Random Forest Classifier has been used to classify UI elements into different categories (e.g., buttons, app bars, cards) based on their features.

For the model training purpose, firstly, we uploaded all the required datasets (the train.csv file and relevant image files) into a Google Drive folder and started implementing the model by importing the datasets into the created Google Colab Notebook.

We broke down each textual guideline into a functional behavior that can be directly executed within the code. Lambda functions are used because they provide a concise way to implement small, single-purpose logic that aligns with each guideline, ensuring modularity and reusability. This approach simplifies the process of evaluating or enforcing specific design rules programmatically, making the implementation both efficient and adaptable. The color guidelines that has been converted into Lambda functions are as below (Figure 7).

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

```
guidelines = {
    'APP_BAR': [primary_color],
    'BACKDROP_BACK': [primary_color],
    'BACKDROP_FRONT': [[255, 255, 255]],
    'SHEET_SURFACE': [[255, 255, 255]],
    'MODAL_SHEET': [[255, 255, 255], primary_color, secondary_color],
    'CARD': [[255, 255, 255]],
    'BUTTON': [primary_color],
    'RADIO_BUTTON': [[255, 255, 255]],
    'CHECK_BOX': [[255, 255, 255]],
    'FAB': [secondary_color],
    'SELECTION_CONTROL': [secondary_color],
    'ICON': [primary_color],
    'TEXT_INPUT': [secondary_color],
    'LABEL': [primary_color]
}
```

Fig 7: The Imported JSON Data

As the annotated data has only the UI element ID column and an annotated data column containing element location coordinates and element description (Figure 6), we had to create three more columns called is_compliant, what_complaient, and improvement. These columns contain if the UI design follows the guidelines (is_compliant), if not, what the violated guidelines (what_compliant), and how a designer can improve the design according to the guidelines (improvements). The output CSV file preview is available under Figure 8. After attaching those data, we generated another CSV file, and the rest of the things have been implemented using that CSV file.

A	B	C	D	E	F	G	H	I	J
image_path	element_type	label	actual_color	expected_color	compli	improvements			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.",			
/content/drive/I	LABEL	LABEL	#eefafa	#e7e8e8	TRUE	["Ensure ICON 'ICON' uses the color #e7e8e8.", "E			

Fig 8: Generated CSV File with New Columns - Part of the Dataset

Using the random forest classifier, we could train the model. Finally, we created the model pickle file. We have selected the random forest classifier over other related machine learning models.

The researchers followed several steps during the model creation phase before finalizing the model.

The first step is to mark the extracted element annotations on each UI, as shown in Figure 09.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

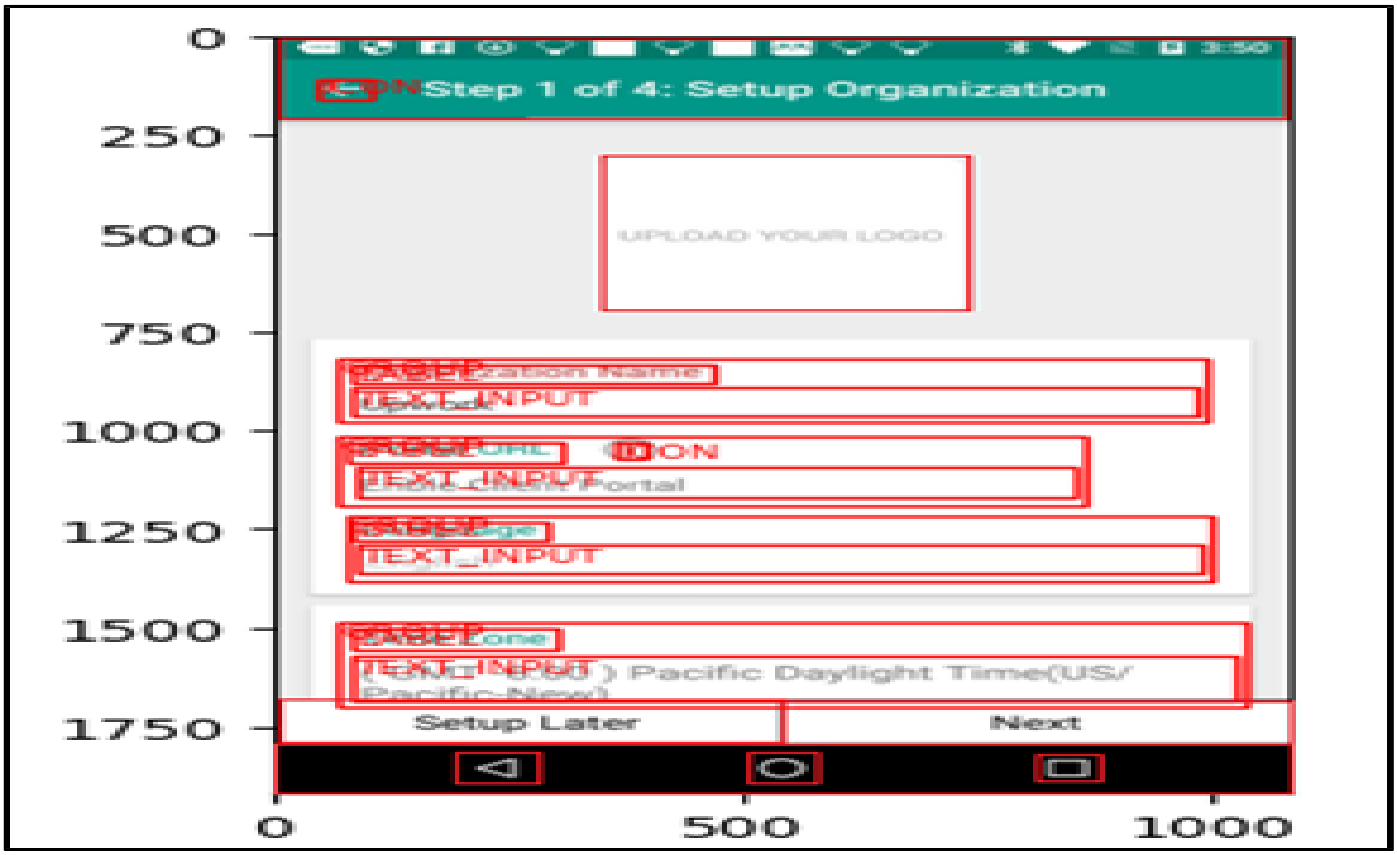

Fig 9: Extracted Annotations Marked on the UIs (Example)

The marked elements will be cropped in the second step to identify each element's most dominant color (Figure 10).

Fig 10: Cropped UI Elements and Dominant Color Identification

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

After making a comparison between the identified primary and secondary colors for the whole UI, identifying each element's dominant color as well as the UI design guidelines, it could generate a set of data that contains element_type, actual_color, expected_color, and improvements according to the UI design guidelines as mentioned in figure 11.

	element_type	label	actual_color	expected_color	compliant	improvements
0	ICON	ICON	#079489	#e7e8e8	False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
1	ICON	ICON	#898989	#e7e8e8	False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
2	TEXT	TEXT_INPUT	#c2c2c2		False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
3	TEXT	TEXT_INPUT	#e8e8e8		False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
4	TEXT	TEXT_INPUT	#e3e3e3		False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
5	TEXT	TEXT_INPUT	#858585		False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
6	LABEL	LABEL	#2a9385	#e7e8e8	False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
7	LABEL	LABEL	#299486	#e7e8e8	False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
8	LABEL	LABEL	#eefafa	#e7e8e8	True	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
9	LABEL	LABEL	#b55355	#e7e8e8	False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...
10	GROUP	GROUP	#f2eceb		False	[Ensure ICON 'ICON' uses the color #e7e8e8., E...

Fig 11: Generation of Training Data (Example)

Then, all the training data has been transferred into a CSV file, the RandomForestClassifier was used to classify image data and create the model (Figure 12).

```
# split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# train the random forest model
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Fig 12: Usage of Random Forest Classifier

Finally, it could check the accuracy as well as generate a classification report, as mentioned in Figure 13. The evaluation shows that, overall, the classification model achieves a high accuracy of 98.26%, but this is significantly biased toward the majority class (class 0). Though class 0 has perfect precision, recall, and F1-score, the minority class performs very poorly, with class 1 having a precision of 41%, recall of 56%, and an F1-score of 47%. This is because of the significantly higher number of samples in class 0 compared to class 1 (3844 versus 317). While high weighted average metrics due to the dominance of class 0 are nice, the low performance for class 1 suggests that further improvements are needed-such as dataset balancing, focusing on recall or F1-score for evaluation, or adjusting the model's decision threshold.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')

print('Classification Report:')
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9826231286613149
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      3844
           1       0.41      0.56      0.47       317

    accuracy                           0.98      4161
   macro avg       0.71      0.78      0.73      4161
weighted avg       0.98      0.99      0.99      4161
```

Fig 13: Accuracy and Classification Reports for the Trained Model

D. Frontend Development Of Colorguard Web Application

The VisualStudio Code has been used as the IDE for frontend development purposes. The frontend has been developed using HTML5, PHP, and Python. PHP handled HTTP requests, allowing the frontend to communicate with the backend. The user interface included components for uploading images. Once user upload an image to the ColorGuard web application, results were retrieved from the backend and displayed in a user-friendly format. The frontend facilitated a seamless experience for users to upload screenshots, view detected UI elements, and receive compliance feedback in real-time.

The MVC (Modal – View – Controller) architecture has been followed to implement the project in order to achieve a clearer implementation process.

From the users' point of view, the web application should be straightforward and user-friendly to ensure ease with which users can upload screenshots for analysis and annotation. For that reason, the major interface in this web application is a form whereby users can select an image file from their devices and submit it for validation against the design guidelines. The layout is neat and very user-friendly, bringing clarity to the users when uploading their screenshots (Figure 14).

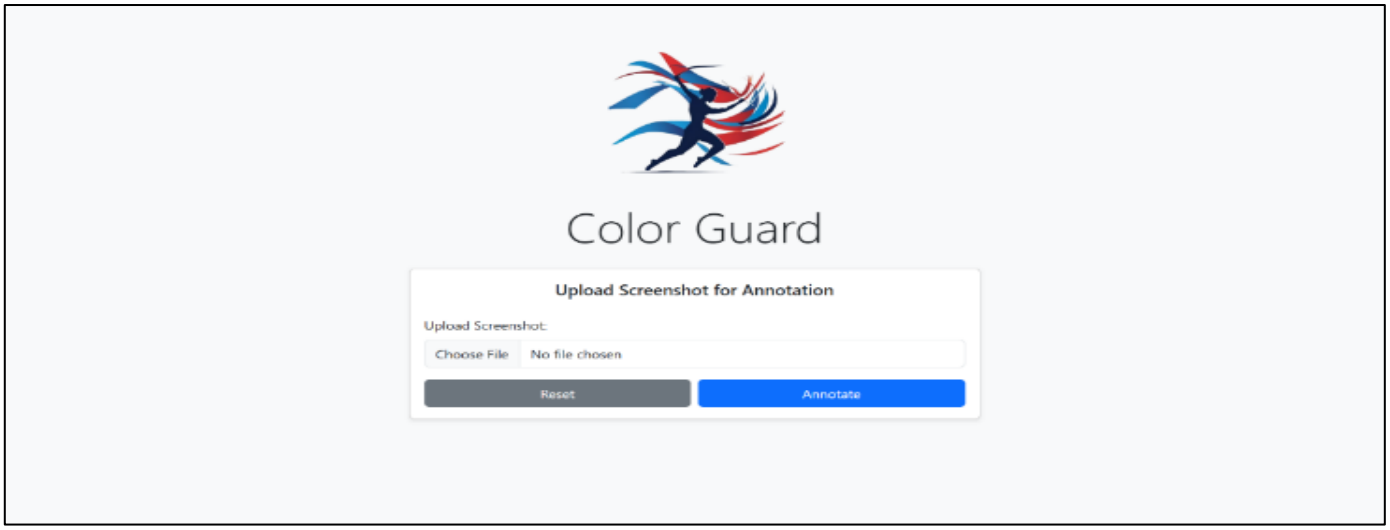

Fig 14: Image Upload Screen of Color Guard

For the development of the frontend, HTML5 has been used for structuring content, thus guaranteeing compatibility over a wide range of web browsers. A responsive design approach has been implemented and incorporates the use of viewport meta tags that allow the application to adapt seamlessly to different screen sizes concerning desktops, tablets, and mobile devices. It grants access to and efficient usage of this application on any device.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

It is an interface form that allows one to upload images. These have been sent to the backend over HTTP POST requests. This communication between the frontend and the backend will enable users with the capability to submit their screenshots for analysis. Upon submitting the screenshot, this system processes the image and responds with feedback if it follows specific guidelines such as those by Google's Material Design standards (Figure 15).

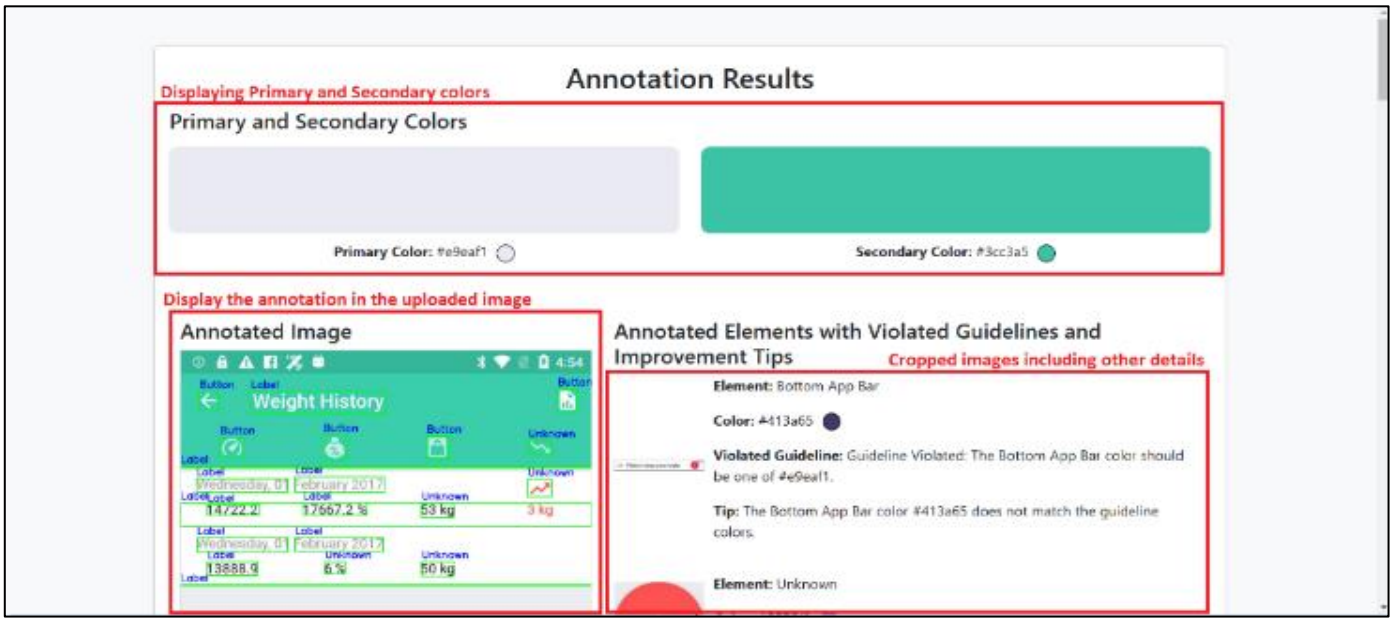

Fig 15: After Annotating the Image (The Final Output of the ColorGuard Application)

When developing the frontend, much attention was paid to creating a smooth user experience. The user is guided through a very intuitively simple process of uploading their screenshots and getting instant feedback without overwhelming them with details they do not want to know. Moreover, the lightweight design improves the application's performance, especially when users are on poor bandwidth.

E. Backend Development

The backend of the ColorGuard was developed using Python. Image processing was performed using OpenCV and the ColorThief library to extract dominant colors from the uploaded images. ColorThief library is handy for projects where you need to analyze the color composition of images, as it can identify the most prominent colors efficiently. This project uses the ColorThief library to extract the dominant colors from a UI image.

OpenCV has been integrated to identify UI elements within a screenshot without predefined coordinates. NumPy was utilized for numerical operations, such as color comparison, while Pandas was used for data manipulation and analysis. The trained Random Forest model was integrated into the backend to predict compliance. Multiprocessing was utilized to handle large datasets efficiently, enhancing the backend performance.

F. How to use the Application

First, the novice designer must upload their UI design, which needs to be checked with the ColorGuard application from the main screen (Figure 14). When the user presses the 'Annotate' button, they can see the annotation results and compliance details, including violated UI design guidelines, on the second screen (Figure 15).

V. RESULTS AND DISCUSSION

A structured process of testing was therefore required, with the involvement of human experts who could make a valid evaluation of the ColorGuard application in picking appropriate colors for UI designs, as opposed to human-selected color preferences for similar UI designs.

A dedicated web application was developed to facilitate the collection of data from human experts (Figure 16). The application provided the following functionalities:

- Image Uploading: Experts were allowed to upload UI design images.
- Primary and Secondary Color Selection: Experts selected the dominant (primary) and complementary (secondary) colors for the uploaded images.
- Element-by-Element Color Selection: For each UI element within the image, experts selected their preferred colors based on their professional judgment.

This system streamlined the data collection process and allowed experts to provide detailed input for each UI design.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

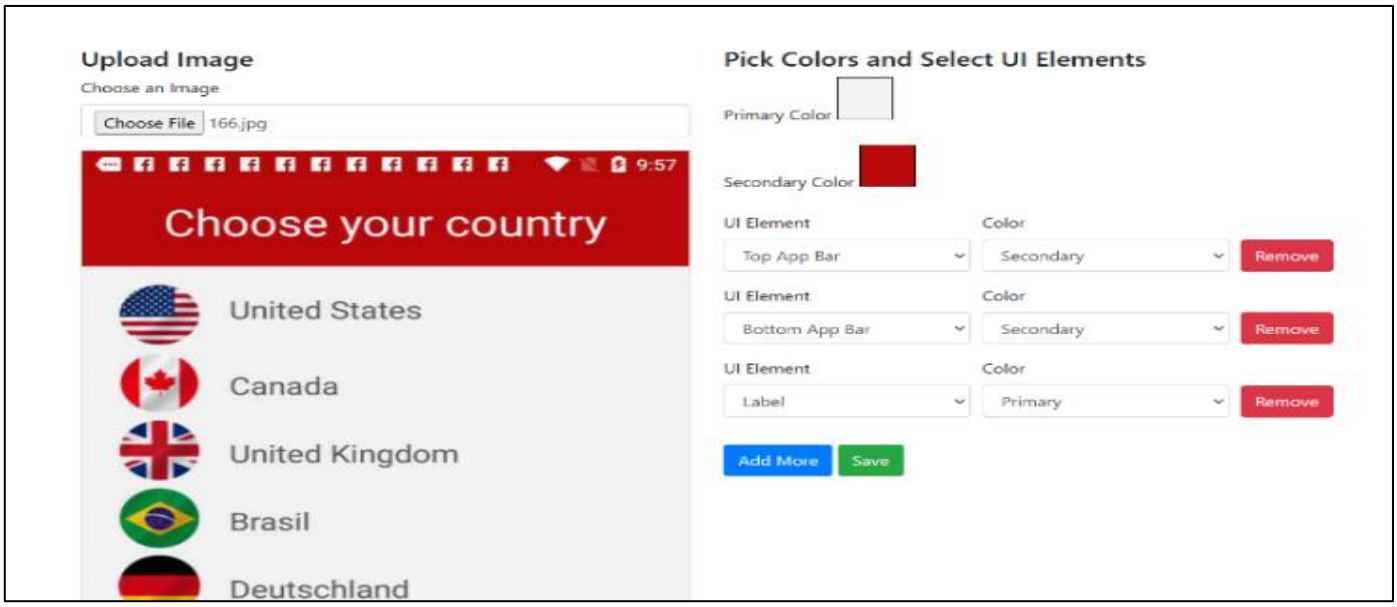

Fig 16: The Feedback-Collecting Web Application Home Page

This research engaged five human experts from a well-reputed software company with 3 – 5 years of experience in UI/UX designing and Software Engineering. Each of the human experts was requested to evaluate ten different UI images. This resulted in a detailed color preference dataset on 50 UI designs, which can be used to benchmark the performance of the ColorGuard application.

In order to provide a wide range of UI design coverage during the testing process, a wide variety of screenshots were chosen. These ranged from simple pages containing just a few elements, such as login screens, to more complex dashboard-type pages with many buttons, labels, and navigational components. Such variability in image complexity was very important for the test with respect to assessing the tool's ability to cope with different design contexts, including but not limited to cluttered versus minimal interfaces. The images were also selected to vary their color palette so that dominant and subtle color changes could be well tested in the system.

The ColorGuard application's outputs were systematically compared to inputs provided by human experts to evaluate the accuracy of its output. This comparison focused on two key dimensions: how well the application's primary and secondary color choices align with those selected by experts and to what extent the application's suggested colors for individual UI elements match the experts' preferred colors. These comparisons highlighted how well the application could produce designs that captured human design sensibilities and where its potential for improvement may lie.

Experts' feedback		
image_id	ui_element	color
52	Top App Bar	Primary
52	Bottom App Bar	Secondary
52	Label	#000000
52	Button	Secondary
52	Button	Secondary
52	Button	Secondary
0	Top App Bar	Primary
0	Bottom App Bar	Primary
0	Label	Secondary
0	Label	Secondary
0	Label	Secondary
0	Button	Primary
0	Button	Primary
0	Button	Primary
0	Button	Primary
0	Button	Primary
0	Button	Primary
3	Top App Bar	Primary
3	Bottom App Bar	Primary

ColorGuard response		
image_id	ui_element	color
52	Top App Bar	Primary
52	Bottom App Bar	Secondary
52	Label	Primary
52	Button	Secondary
52	Button	Primary
52	Button	Secondary
0	Top App Bar	Primary
0	Bottom App Bar	Primary
0	Label	Secondary
0	Label	Secondary
0	Label	Secondary
0	Button	#0000
0	Button	Primary
0	Button	Primary
0	Button	Primary
0	Button	Primary
0	Button	Primary
3	Top App Bar	Secondary
3	Bottom App Bar	Primary

Fig 17: Comparison between the outputs of experts' Feedback and Color Guard

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

This approach quantified how well the ColorGuard application conformed to the human sensibilities of design. Furthermore, it revealed some points where the preferences of an expert for better usability and aesthetic appeal in UI design can further tune automation.

VI. LIMITATION

Testing revealed limitations in the automated annotation tool, including difficulty distinguishing visually similar elements with different functions, such as navigation links and buttons, and challenges extracting exact colors from elements with gradients or transparent backgrounds. Additionally, reliance on size and aspect ratio for classification led to misidentifications in specific UI designs. These issues highlight areas for further improvement.

- *Availability of Large Visual Models*

While powerful models like Contrastive Language–Image Pre-training (CLIP) [34], Self-Distillation with No Labels (DINO) [35], and Segment Anything Model (SAM) [36] have become available, they were not integrated into mainstream workflows when this research began. These models could significantly enhance object detection, segmentation, and element classification tasks. However, the current approach relies on traditional computer vision techniques and rule-based logic, which may not fully leverage state-of-the-art advancements. Future work will explore incorporating these models to improve the system's accuracy and versatility.

- *Misclassification of Similar Elements*

The contour-based classification model struggled to differentiate GUI elements with similar appearances but distinct functions, such as misclassifying navigation links as buttons or failing to distinguish between labels and text fields. This issue is particularly prevalent in UI designs where components differ subtly in appearance or are defined by interaction rather than visual cues. The reliance on hard-coded rules, including element size, aspect ratio, and position, limits the model's adaptability, making it prone to errors in unconventional UI patterns. Incorporating machine learning could offer a more flexible and efficient solution for handling these edge cases.

- *Dependence on Image Quality and Preprocessing*

The quality of input images heavily influences the tool's accuracy. Low-resolution or highly compressed screenshots often result in poor contour detection, leading to missed or incorrectly recognized elements. Additionally, the pre-processing steps, such as thresholding and morphological transformations, also rely on specific parameter tuning, which may not generalize effectively across different image types.

FUTURE WORK

Future iterations of this project will enhance functionality by focusing on generating image-based improvement suggestions and offering modified UI designs. Advanced image processing and computer vision techniques, such as Generative Adversarial Networks (GANs), will be explored to propose compliant UI modifications [37]. By training GANs on datasets of compliant designs, the system will automatically generate visual recommendations that align with design standards, simplifying the improvement process.

The tool will introduce real-time compliance checking, providing immediate feedback to designers for on-the-fly adjustments. Advanced customization options will enhance adaptability, allowing users to tailor the tool to specific needs. Expanding support to frameworks like Apple's Human Interface Guidelines and Microsoft's Fluent Design System will increase its versatility across diverse projects.

AI and machine learning will enable context-aware design suggestions, improving compliance accuracy through adaptive learning from user feedback and usage data. Collaborative features will facilitate team sharing of designs and reports, enhancing communication and iteration. Analytics tools will track compliance trends and improvements, helping users identify and address common issues effectively.

Integrating popular design platforms like Adobe XD, Figma, Sketch, and a mobile app for on-the-go compliance checking will improve accessibility. Educational resources, tutorials, and workshops will empower users to utilize the tool's capabilities fully. Community-driven initiatives, such as open-sourcing parts of the project, will encourage innovation and user contributions.

Regular updates aligned with design trends and a feedback loop will keep the tool cutting-edge. These enhancements will transform it into a comprehensive design assistant, streamlining the UI/UX design process and supporting the creation of high-quality, guideline-compliant interfaces.

ACKNOWLEDGMENT

We express our gratitude to our advisor for their invaluable guidance and support, our colleagues for their constructive feedback, and the support received. Special thanks to those who assisted with data collection and analysis and to our families and friends for their unwavering encouragement throughout this research.

REFERENCES

[1]. Sambin, G., 2023. Usability of Safety Critical Applications in Enterprise Environments: Defining Guidelines for Error Preventing UI/UX Patterns and Improving Existing Interfaces (Doctoral dissertation, Politecnico di Torino).

[2]. Android Developers. (n.d.). Accessibility | Mobile. [online] Available at:https://developer.android.com/design/ui/mobile/guides/foundations/accessibility.

[3]. Guo, S., Shi, Y., Xiao, P., Fu, Y., Lin, J., Zeng, W. and Lee, T.Y., 2023. Creative and progressive

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

interior color design with eye-tracked user preference. ACM transactions on computer-human interaction, 30(1), pp.1-31.

[4]. Yang, B., Xing, Z., Xia, X., Chen, C., Ye, D. and Li, S., 2021, May. UIS-hunter: Detecting UI design smells in Android apps. In 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion) (pp. 89-92). IEEE.

[5]. Liu, Z., Chen, C., Wang, J., Huang, Y., Hu, J. and Wang, Q., 2020, December. Owl eyes: Spotting UI display issues via visual understanding. In Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering (pp. 398-409).

[6]. Chen, J., Chen, C., Xing, Z., Xia, X., Zhu, L., Grundy, J. and Wang, J., 2020. Wireframe based UI design search through image autoencoder. ACM Transactions on Software Engineering and Methodology (TOSEM), 29(3), pp.1-31.

[7]. Unger, R. and Chandler, C., 2023. A Project Guide to UX Design: For user experience designers in the field or in the making. New Riders.

[8]. Aczel, B., Szaszi, B. and Holcombe, A.O., 2021. A billion-dollar donation: estimating the cost of researchers' time spent on peer review. Research Integrity and Peer Review, 6, pp.1-8.

[9]. Sabale, M.M., Pande, V.A., Tagalpallewar, A.A., Swami, A.G., Pawar, A.T. and Baheti, A.M., 2024. Maintaining data safety and accuracy through data integrity (DI): A comprehensive review. Research Journal of Pharmacy and Technology, 17(5), pp.2431-2440.

[10]. Zhao, T., Chen, C., Liu, Y. and Zhu, X., 2021, May. Guigan: Learning to generate GUI designs using generative adversarial networks. In 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE) (pp. 748-760). IEEE

[11]. Ajiga, D., Okeleke, P.A., Folorunsho, S.O. and Ezeigweneme, C., 2024. The role of software automation in improving industrial operations and efficiency. International Journal of Engineering Research Updates, 7(1), pp.22-35.

[12]. Viswanathan, S., & Peters, J., 2010. Automating UI guidelines verification by leveraging pattern based UI and model based development. CHI '10 Extended Abstracts on Human Factors in Computing Systems. https://doi.org/10.1145/1753846.1754222.

[13]. Shneiderman, B., 2020. Human-centered artificial intelligence: Reliable, safe & trustworthy. International Journal of Human–Computer Interaction, 36(6), pp.495-504.

[14]. Amor, R. and Dimyadi, J., 2021. The promise of automated compliance checking. Developments in the built environment, 5, p.100039.

[15]. Wright, A., Wang, Z., Park, H., Guo, G., Sperrle, F., El-Assady, M., Endert, A., Keim, D., & Chau, D., 2020. A Comparative Analysis of Industry Human-AI Interaction Guidelines. ArXiv, abs/2010.11761.

[16]. D. L. Stone, User interface design and evaluation. Amsterdam: Elsevier, 2009.

[17]. Hamidli, N., 2023. Introduction to UI/UX design: key concepts and principles. Academia. URL: https://www. academia. edu/98036432/Introduction_to_UI_UX_Design_Key_Concepts_and_Principles [accessed 2024-04-27].Journal Name:ACM Trans. Graph.

[18]. Lokman, A.M. and Hussin, S.N., 2011. Kansei website interface design: Practicality and accuracy of Kansei Web Design Guideline. In 2nd International Conference on User Science and Engineering (i-USEr).

[19]. Material Design. (n.d.). Material Design. [online] Available at: https://m2.material.io/design/color/the-color-system.html#color-theme-creation.

[20]. Whitaker, R., 2020. iOS Accessibility Features – General. , pp. 175-201. https://doi.org/10.1007/978-1-4842-5814-9_7.

[21]. Wood, T., 2011. Putting the iOS Human Interface Guidelines in Context., pp. 1-6. https://doi.org/10.1007/978-1-4302-3880-5_1.

[22]. Hu, P., Ma, P., & Chau, P., 1999. Evaluation of user interface designs for information retrieval systems: a computer-based experiment. Decis. Support Syst., 27, pp. 125-143. https://doi.org/10.1016/S0167-9236(99)00040-8.

[23]. Awwad, A., Schindler, C., Luhana, K., Ali, Z., & Spieler, B., 2017. Improving pocket paint usability via material design compliance and internationalization & localization support on application level. Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services. https://doi.org/10.1145/3098279.3122142.

[24]. Kalac, E., Borovina, N., & Boskovic, D., 2021. Preserving interaction design principles while implementing Material Design Guidelines. 2021 20th International Symposium INFOTEH-JAHORINA (INFOTEH), pp. 1-6. https://doi.org/10.1109/INFOTEH51037.2021.9400523.

[25]. Jung, D. and Kim, S.I., 2020. A Study on Mobile Application UI Design Components & Design Guidelines-Focused on the Google Material Design Guidelines. Journal of Digital Convergence, 18(5), pp.417-423.

[26]. Bessa, M., Bostanabad, R., Liu, Z., Hu, A., Apley, D., Brinson, C., Chen, W., & Liu, W., 2017. A framework for data-driven analysis of materials under uncertainty: Countering the curse of dimensionality. Computer Methods in Applied Mechanics and Engineering, 320, pp. 633-667. https://doi.org/10.1016/J.CMA.2017.03.037.

[27]. Adekotujo, A., Odumabo, A., Adedokun, A. and Aiyeniko, O., 2020. A comparative study of operating systems: Case of windows, unix, linux, mac, android and ios. International Journal of Computer Applications, 176(39), pp.16-23.

[28]. Kalac, E., Borovina, N., & Boskovic, D., 2021. Preserving interaction design principles while implementing Material Design Guidelines. 2021 20th International Symposium INFOTEH-JAHORINA

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1435

(INFOTEH), pp. 1-6. https://doi.org/10.1109/INFOTEH51037.2021.9400523.

[29]. Tilstra, A., Backlund, P., Seepersad, C., & Wood, K., 2015. Principles for designing products with flexibility for future evolution. International Journal of Mass Customisation. https://doi.org/10.1504/IJMASSC.2015.069597.

[30]. Park, W., Han, S., Kang, S., Park, Y., & Chun, J., 2011. A factor combination approach to developing style guides for mobile phone user interface. International Journal of Industrial Ergonomics, 41, pp. 536-545. https://doi.org/10.1016/J.ERGON.2011.04.002.

[31]. Visnapuu, K., 2023. Consistency, Efficiency, and Scalability in Design Systems: A Comparative Analysis of Material Design, Apple's Human Interface Guidelines, and IBM Design Systems.

[32]. Onur Gunes (2021). RICO dataset. [online] Kaggle.com. Available at: https://www.kaggle.com/datasets/onurgunes1993/rico-dataset.

[33]. GitHub. (2024). google-research-datasets/rico_semantics. [online] Available at: https://github.com/google-research-datasets/rico_semantics.

[34]. Pan, X., Ye, T., Han, D., Song, S. and Huang, G., 2022. Contrastive language-image pre-training with knowledge graphs. Advances in Neural Information Processing Systems, 35, pp.22895-22910.

[35]. Das, R. and Sanghavi, S., 2023, July. Understanding self-distillation in the presence of label noise. In International Conference on Machine Learning (pp. 7102-7140). PMLR.

[36]. Zhang, C., Liu, L., Cui, Y., Huang, G., Lin, W., Yang, Y. and Hu, Y., 2023. A comprehensive survey on segment anything model for vision and beyond. arXiv preprint arXiv:2305.08196.

[37]. Zhao, T., Chen, C., Liu, Y. and Zhu, X. (2021). GUIGAN: Learning to Generate GUI Designs Using Generative Adversarial Networks. [online] IEEE Xplore. doi:https://doi.org/10.1109/ICSE43902.2021.00074.

SAJANI ANUPAMA BALASOORIYA is an Academic Instructor in the Department of Information Technology at the Faculty of Computing of Sri Lanka Institute of Information Technology (SLIIT). She holds a B.Sc. (Hons.) in IT from SLIIT and completed M.Sc. IT at the same university. Her research interests are for Human-Computer Interaction (HCI) and Machine Learning. She can be reached via email anubalasooriya2017@gmail.com.

JAGATH WICKRAMARATHNE, a Senior Lecturer in the Department of Software Engineering at the Faculty of Computing, Sri Lanka Institute of Information Technology (SLIIT). He holds an MSc from the University of Moratuwa, Sri Lanka. His research interests encompass Human-Computer Interaction (HCI) and User Experience Design (UXD), with particular focus on improving user experience and retention. Mr. Wickramarathne has published many research papers in collaboration in these areas, demonstrating his extensive expertise and contributions of the field. He is a member of IEEE. He can be reached via email at jagath.w@sliit.lk.