INFORMATICS INSTITUTE OF TECHNOLOGY

In Collaboration with

UNIVERSITY OF WESTMINSTER

CREATEARO - Automating Violation Detection on iOS UI

A Final Project Report (FPR) by

Miss. Chavika Kodithuwakku

Supervised by

Mr. Asanjay Fernando

Submitted of the requirements for the
BEng in Software Engineering degree at the University of Westminster.

Declaration

I hereby certify that this dissertation and all of its linked subcomponents are my original work and have not been submitted/presented nor are those who are currently being submitted/presented as content for any degree or other qualification program at any other university or institution. Facts extracted from credible external sources have been cited and appropriately credited.

Name of the student : Chavika Ravihari Wimalaguna Kodithuwakku

Registration Number : 2018468/ w1742135

chavika

Signature :

Date : June 9, 2022

Abstract

The term Graphical User Interface (GUI) is also used for indicating UI. Basically, the role of a GUI is to function as an interacting media in between the functions of the user application and the user. The GUI is the only visible part of an application to a user. The user is driven to the final purpose of the application through these interfaces.

That is why designing the UI of an application becomes the most critical step when developing an application with the target of adding the application to an application store.

In that case, it does not matter when it comes to the common law in the guidelines. Suppose the app is not created according to standards releasing the app to the relevant app stores. In such a case, there is a high chance of being rejected. That is because any user is expecting to reach their goals soon. And if the users must use their own effort to identify the things in the interface, the user thinks twice before using the app again.

So, there is no point in the uniqueness of the app's goal if it provides a bad experience to a user; they do not think to use it again. This will eventually increase the speed of uninstalling the app compared to installing it because of this bad experience.

The **CREATEARO - Automating Violation Detection on iOS UI** system helps to avoid this situation.

Keywords : iOS violation detection system; Component Detection; Text Detection; Computer Vision; Human Interface Guidelines; Human Computer Interaction

Acknowledgement

Beginning with merely identifying a gap in a completely unfamiliar domain, I am privileged to have conducted a successful study that filled the gap. This would not have been possible without the immense guidance, instructions, and knowledge provided by the numerous people I met on this incredible journey. My supervisor, Mr. Asanjay Fernando, however, stood out among all of these individuals. If my research were a tall building, he would be its sturdy foundation. During the stormy seasons, the structure would have collapsed into a thousand pieces if not for him. I was able to conduct a proper research due to his reviews, advice, and perspectives on research, as well as all the methodologies he required me to use. A special thank you goes out to the industry and academic experts who participated in requirement gathering and evaluation interviews for this research project, giving up their valuable time to listen to my research presentations and provide feedback. This greatly assisted me in making this research a success. I would like to thank all my professors, the management of the Informatics Institute of Technology, and my fellow colleagues for their support throughout my senior project. Lastly, but most importantly, I would like to thank my parents, who have been the most encouraging and nurturing people in my life, always willing to provide anything to help me achieve success.

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 Chapter Overview

In this proposal, a tool that can auto-detect User Interface (UI) violations and human interface guidelines in iOS with Human Interface Guidelines (HIG) principles is discussed.
The introduction chapter gives a detailed overview of the entire research project.

In conclusion, the documentation clearly explains the problem, its research relevance, and the author's motivation for doing the study.

1.2 Problem Domain

When developing an application, it is essential to focus on both the functional development as well as the UI.
The term Graphical User Interface (GUI) is also used for indicating UI. Basically, the role of a GUI is to function as an interacting media in between the functions of the user application and the user. The GUI is the only visible part of an application to a user. The user is driven to the final purpose of the application through these interfaces.
That is why designing the UI of an application becomes the most critical step when developing an application with the target of adding the application to an application store.
In that case, it does not matter when it comes to the common law in the guidelines. Suppose the app is not created according to standards releasing the app to the relevant app stores. In such a case, there is a high chance of being rejected. That is because any user is expecting to reach their goals soon. And if the users must use their own effort to identify the things in the interface, the user thinks twice before using the app again.
So, there is no point in the uniqueness of the app's goal if it provides a bad experience to a user; they do not think to use it again. This will eventually increase the speed of uninstalling the app compared to installing it because of this bad experience.
Therefore, it is better to strictly consider the function requirement and the quality of the interface.
There are 2 key roles of a GUI as follows:

1.Assisting the user in the identification of the functions of the application

2.Driving the user to the final purpose of using the application through the user flow A UI can be divided into elements.

Any element of a UI can be categorized under one component from the following 3 components.

1. Input Controls
2. Navigational Components
3. Informational Components

In this paper, the general components that are used in almost all UIs relevant for the abovementioned main components are explained. Moreover, the Design Aspects and Design Dimensions of each and every component are discussed.

10 major UI components that are in general use are explained in detail as follows:

1. App Bars: Bottom
2. App Bars: Top
3. Bottom Navigation
4. Buttons
5. Floating Action Buttons
6. Lists
7. Navigation Drawer
8. Text Fields
9. Tabs
10. Banners

There are four major design aspects as follows:

1. Anatomy
2. Placement
3. Behaviors
4. Usage

As this paper focuses on the guidelines of the UIs that do not have animations; when discussing the dimensions, it can be divided into-:

1. Typography

2. Iconography
3. Shapes
4. Colors

1.3 Problem Definition

By focusing on UI components, their aspects, and various dimensions, providing an application to a user which gives the best experience for the user's purpose is important.

If one of these aspects gets violated, the application may offer a bad experience to the user.

Consequently, this may result in the rejection of the application by the user.

Therefore, it is pretty important to focus more on these UI components when developing applications.

Nowadays, there is a huge amount of resources available for studying the field of application development. As a result, many people release new applications.

But due to the violation of UI principles and Human-Computer Interaction (HCI) principles from these applications, there is considerable potential for uninstalling such applications by the user.

Normally users do not like to learn something and test it. Instead, users prefer to carry on things in the usual way to reach their purpose.

If there is no good connection between the user and functionalities, the application may be considered a failure by the user even though it comprises great functionalities.

No matter which platform is used to develop the application, which means cross-platform or hybrid, or whatever platform; it is essential to pay considerable attention to UI before developing an application and releasing it to a user base.

1.3.1 Problem Statement

Because of the HIG and UI principle violations that happened by some developers in UI development in an iOS application, the user base experiences a bad experience, and the users tend to uninstall apps.

1.4 Research Motivation

There was no one in the industry where the author interned who understood UI concepts very well, and the author had before learned to create applications by following tutorials but had not paid enough attention to UI principles.

Due to these reasons, the author had to face many problems while working as a front-end developer there. As a result, the author had to learn UI principles from the start and create an extra effect for it.
Therefore, the author considered developing a tool for detecting violations in iOS applications.

1.5 Existing work

Advantage	Limitation	Citation
Handles a significant, though, and very practical problem with amazing efficiency and precision, and is both useful and scalable in the eyes of industrial designers and developers.	Currently, the tool imposes minor constraints on designers creating Sketch mock-ups, the most important of which is that component bounding boxes do not overlap.	(Moran, May 2018)
These models may be used to search for UI design samples based on user-provided natural language descriptions and to produce professional-looking UI designs from simpler, less elegant design drafts.	The core concept is to create and deploy sophisticated deep learning models based on recurrent neural networks and generative adversarial networks to learn UI design patterns from millions of existing mobile apps. This is only for design patterns; no idea about how components work.	(Nguyen, May 2018)
Analyzing a huge number of class files from four big Java open-source software projects yielded software metrics and the detection of design smells.	Determine the tool's accuracy and recall in addition, we intend to develop and test the performance of the data mining and Spectral Clustering modules.	(Imran, Dec 2019)
Model-driven educational environments that use simulation as a learning aid by creating a user interface and the underlying program using conceptual domain and presentation models.	Provide feedback-enriched simulation tools to aid in the understanding of UI and application design.	(Ruiz, 2022)

Manual identification of UI design smells is less successful than automatic detection, especially when several pieces of information must be merged, or component-variant-sensitive rules are in there.	According to this, mention only abut android mobile violations. Design systems that define visual do/don't guidelines for a library of UI components in a similar vein, support implicit and de-facto rules. To impose visual design principles in code, eliminate visual-design-related code smells, and improves GUI API design. Should be integrated with design tools to allow just-in-time UI design scent detection.	(Yang, 2021)

Table 1 - Existing work

1.6 Research Gap

There is currently no HIG and HCI-based tool to detect iOS violations. According to the UI principles, you have to check whether the application UI is OK or NOT, after uploading the application to the apple app store. But due to this the software development life cycle (SDLC) has to be re-run when the app is not suitable to publish. And again, this would mean that you have to spend more time and money on these processes.

It is a great advantage to be able to detect violations either in the UI design after the requirement gathering part or during the development process.

1.7 Research Contribution

This report is just for Android UIs. However, this does not apply to iOS. Because the principles that govern each of its components differ somewhat from those of android. Furthermore, in Material Design, the components used in each personas face are not smooth, making accurate detection difficult without a HIG (Human Interface Guidelines).

To overcome this gap, a domain contribution might be made.

1.7.1 Contribution to the Software Engineering domain:

This research will be offering a novel way to identify when to use those components principles and guidelines.

1.7.2 Contribution to the Body of Knowledge:

This research will be used in the domain to the identification of violations from the user interfaces of iOS applications.

1.8 Research Challenge

This research work is mainly based and focused on violations.

There are numerous violations that have been found through this research. It is pretty hard to give examples too.

For example, floating buttons can be considered. The text should be placed at the left side of the icon and that is compulsory.

When considering the difference between,

1. Explicit guidelines
2. Implicit guidelines

Simply differ from how HIG describes it. There are no variations in component design aspects, general design dimensions, or atomic UI information. (Yang, 2021)

As a result, once identified, implicit design principles have the ability to help in the same way that explicit guidelines do. (Yang, 2021)

Depending on the official iOS guidelines, it is possible to derive certain principles from real world applications.

This paper has discovered a kind of inconsistent icon/text usage beyond these guidelines via the observation of the UIs of real applications.

For instance, there are incidents where a user can show feedback on such a flow via an icon. But there might be incidents where a user can show feedback on such a flow via a text. This can be stated as a violation when considering the User Experience. But there is no way to detect it here.

Of course, it is possible to extend this tool for detecting the violations throughout the user flow of an app. It means that this tool can input the overall user experience.

1.9 Research Question

RQ 1: What are the main components that vary the most depending on the user personas of the UIs you are developing or designing?

RQ 2: When developing iOS apps, do you pay close attention to Apple's HIG?

1.10 Aims and Objectives

This research paper will guide designers and developers in avoiding UI violations in iOS applications and releasing a quality product to the app store.

1.11 Project Scope

The project objectives listed below will be used to categorize and explain the research objectives.

1.11.1 In-Scope Aims

1. Through this paper, details that are relevant for each and every component of the iOS Guidelines of HIG are mentioned in detail.
2. The project is about demographic studies regarding HIG. The aspects and dimensions of each component will be discussed.
3. If necessary, a particular component of a UI can be inserted. The violations of that component will be detected without having to insert the complete User Interface.
4. Only the dimensions or aspects of a component can be identified separately.
5. A report of the violations will be generated in real-time.
6. Examples of how the UIs should look without violations will be shown.

1.11.2 Out-Scope Aims

1. Employing Implicit and de-facto guidelines in guidelines.

2. Utilizing development tools such as extensions and plug-ins. Ex: XCode.
3. So, the developer can identify violations while the User Interfaces are being developed in the simulator.
4. To identify if the UIs are user-friendly and detect if they are not.

1.11.3 Diagram showing prototype feature

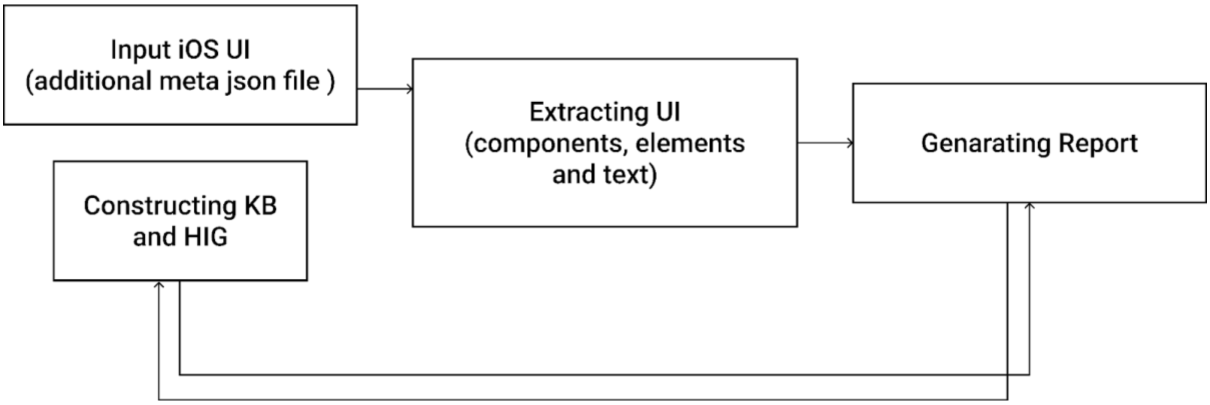


Figure 1 - Diagram showing prototype feature

1.12 Novelty Of Research

The following are the key aims of this research study.

1. Analysis of the reasons behind such violations
2. Showcasing examples to correct such violations just in time after inputting a UI

1.13 Resource Requirements

1.13.1 Hardware Resource Requirements

1. Memory 8 GB 3733 MHz Windows OS.
2. Disk space - Store the dataset.
3. Processor 1.1 GHz Dual-Core Intel Core i3 - A process with multi-threaded performance is good for compiling, data analysis, model training, and data science related tasks.

1.13.2 Software Resource Requirements

1. Python - lightweight programming language with a simple syntax
2. Google Colab - Make use of server computing power
3. PyCharm - A powerful IDE for Python

4. Draw.io - For creating graphs and infographics
5. Mendeley - Citation and reference management software
6. Microsoft Office / Google Docs - Use to manage document submissions
7. GitHub - Manage your source code
8. Google forms - To collect evaluator responses

1.13.3 Skill Requirements

1. Most people have a basic understanding of machine learning algorithms.
2. Understanding of linear regression in statistics
3. Expertise in assessing machine learning-based methods.
4. Skills in writing for a research paper.

1.14 Chapter Summary

The first chapter provided a comprehensive overview of the study's objectives and methods. How the issue area was examined, existing systems were analyzed, as well as how the research gap was developed were detailed in detail in this section. By addressing this issue, we were able to make a positive impact on both the domain and the broader field of software engineering. Additionally, research difficulties, project goals, research objectives, and project scope were outlined. There was also a list of the resources needed to complete the job without difficulty.

CHAPTER 2: LITERATURE REVIEW

2.1 Chapter Overview

It is the purpose of this chapter to map out the concepts and to conduct a literature review and survey of the relevant domain area, systems, approaches, and metrics.
In addition, research gaps in the field will be discussed and explained in this chapter.

2.2 Concept Map

As the early phase, a concept map displaying the scope of the review of the literature was created. This was used to systematically organize and review the literature.

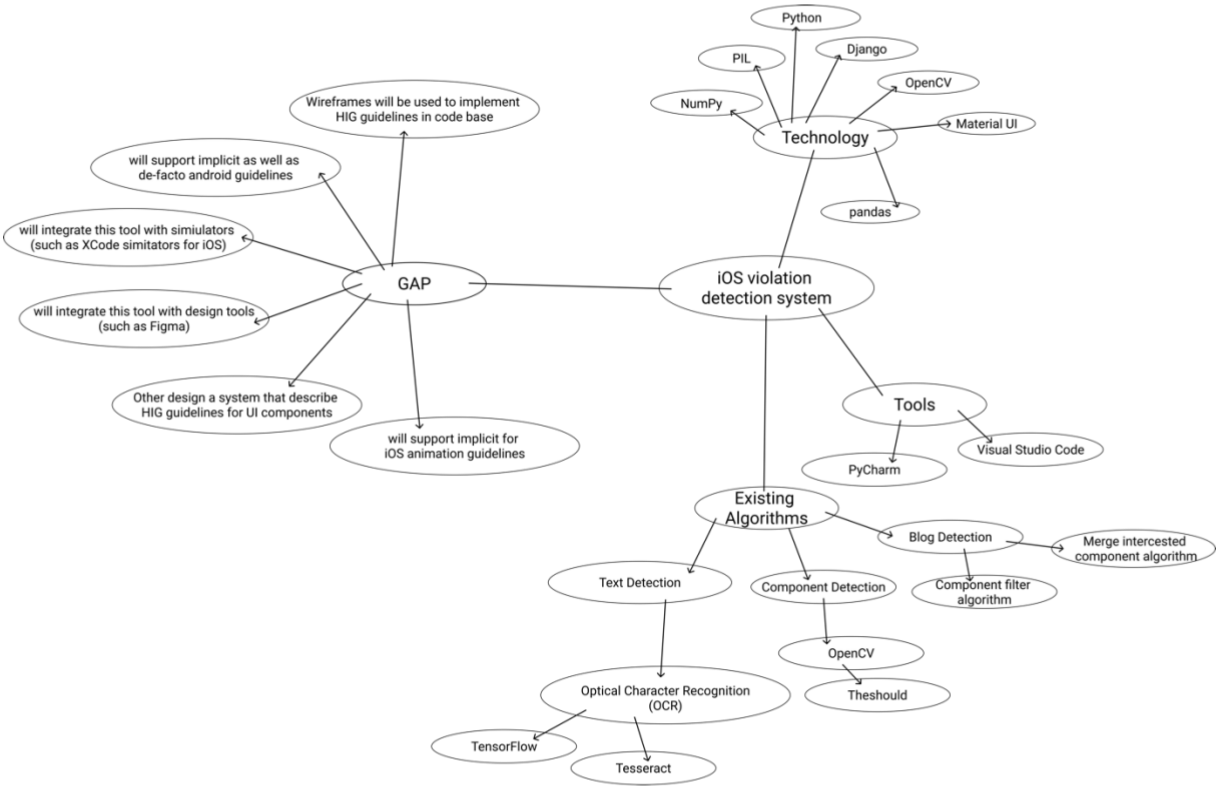

Python
PIL
Django
OpenCV
NumPy
Material UI
Technology
pandas
Wireframes will be used to implement HIG guidelines in code base
will support implicit as well as de-facto android guidelines
will integrate this tool with simiulators (such as XCode simitators for iOS)
GAP
iOS violation detection system
will integrate this tool with design tools (such as Figma)
Other design a system that describe HIG guidelines for UI components
will support implicit for iOS animation guidelines
Tools
Visual Studio Code
PyCharm
Existing Algorithms
Blog Detection
Merge intercested component algorithm
Text Detection
Component Detection
Component filter algorithm
OpenCV
Theshould
Optical Character Recognition (OCR)
TensorFlow
Tesseract

Figure 2 - Concept Map

2.3 Review of the Problem Domain

The purpose of this research is to create a tool to detect violations of iOS UIs. For detecting Violations, here we have created models to detect blocks and, instead of that we have used main models for component detection and text detection.

2.3.1 Introduction to Text Detection

Here this model has used Optical Character Recognition (OCR) system for text detection.

(Hegghammer, 2022) Here we talk about the text detection model as the main detection part, but in reality, we need the text detection model/part to recognize text in detected components. OCR can extract text from an image. Here the model has used a combination of the text detection model and the recognition model to create character recognition.

(Bo Pang, September 10, 2019) First, we used TensorFlow for text detection. But then later switched to Tesseract. Because the main focus here is to recognize the text inside the components. TensorFlow offers an end-to-end open-source ML platform. We can use it with many ML-related concepts in the company of OCR. But as we mentioned in the above author's main purpose in this research is to focus more on text detection and recognition part. So, OCR is used here.

(Chirag Patel, October 2012)

Example:- if there is a "**login**" text inside a button. The main goal is to detect the text and then recognize it as a text.

2.3.2 Introduction to Component Detection

In component detection, we used OpenCV which is a computer vision technique.

(Gary Bradski, Sep 24, 2008) Here all the components in an input UI detect. When we state all the components, not only symbols and elements but texts (characters by characters) are also included. That is why the model mentioned in chapter 2.3.1 is being used.

Since the first stage of this detection model, OpenCV has been used in the system. But that model had some improvements due to the results in different stages. So, the previous parameter is substituted with a new parameter called Min_object_area to increase the number of elements that can be detected by the system. In the first stage, there was a decrease in the quantity of detected components by the 'THESHOULD' value. That means the system missed some small/minute components with UI. Then in the next stage, it needs to be designed/ tuned to detect all the relatable component elements.

And in the horizontal and vertical steps/ stages, the model was designed to add more efficiency.

(Nosal, 14-17 October 2008)

In the implementation stage, this model used Flood fill algorithm. But in the testing phase, Boundary-fill Algorithm was also used. But the Flood fill algorithm which was used in the implementation stage is simpler and more efficient. Because the Flood fill algorithm can process the image containing more than one boundary color.

(S.V.Burtsev, 5, September–October 1993) Therefore, this model used a flood fill algorithm which enabled segmented detection in the final implementation.

2.3.3 Introduction to Blog Detection

The component detection model detects all the components.
As an example:- the component detection model detects components that are inside another component.
Sometimes, those components are detected as single components and also elements detected as blogs by this model. There is no metadata existing model. This is a model which is created by the author. There are some other small functions in this model which helps in segment creation. Example:- if there is a component inside another component it can also be detected by this model. That means all the components are included without any refinement. When that happens these three models are being used for blog detection.
Except that;
Component filter algorithm-: which filters the main components by removing unnecessary components,
Merge intercested component algorithm-: merging two components that are detected as large components if there is any chance.
Here the component filter algorithm is not an existing algorithm. But as an existing algorithm, we can introduce the noise filtering algorithm. (Arya Chowdhury Mugdha, n.d.) (Cheon, 2019.08.31)
But ‘merge intercested component’ also has used the Floodfill algorithm.
And then when the unnecessary components are removed the left components will be updated. And then create detected images connected to the bounding boxes which are related to the updated components.

2.4 Why Detection Models

Mainly this model used component detection to detect elements.
According to the component detection model, all the elements in a UI can be detected. But the quantity of the detected elements is decided based on tuned parameters. Therefore, one character is detected as a single element in a text when detecting elements by the model. That is why this model had to use text detection. Except that blog detection is for detecting all the elements as a group or a batch.

2.5 Proposed Architecture

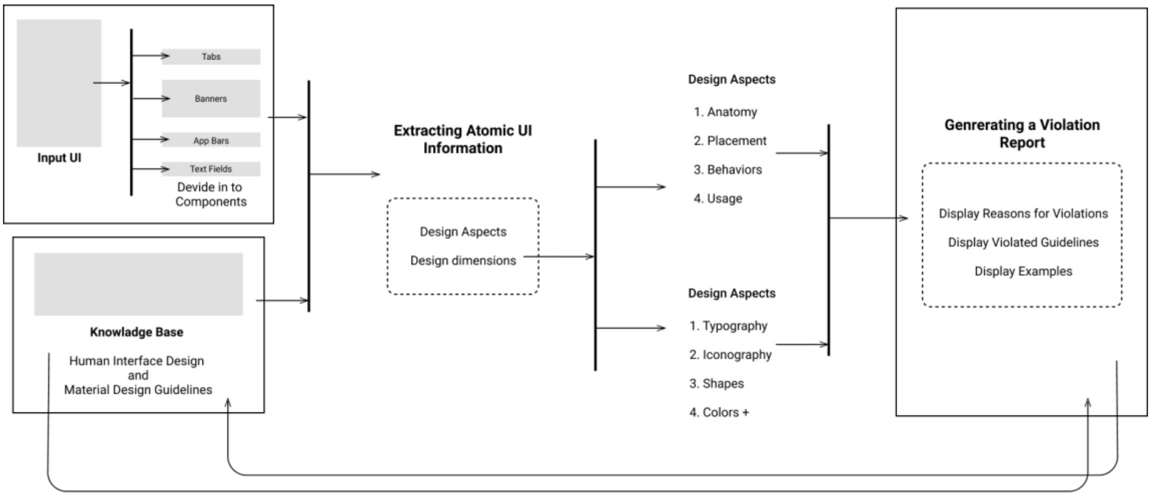


Figure 3 - Proposed Architecture

2.6 Existing Detection System

2.6.1 General Architecture of an iOS violation detection

There is no existing system here, So we can not talk about General Architecture. But as in chapter 2.5 the Proposed Architecture has been applied to the system.

The following have proposed system of an iOS violation detection

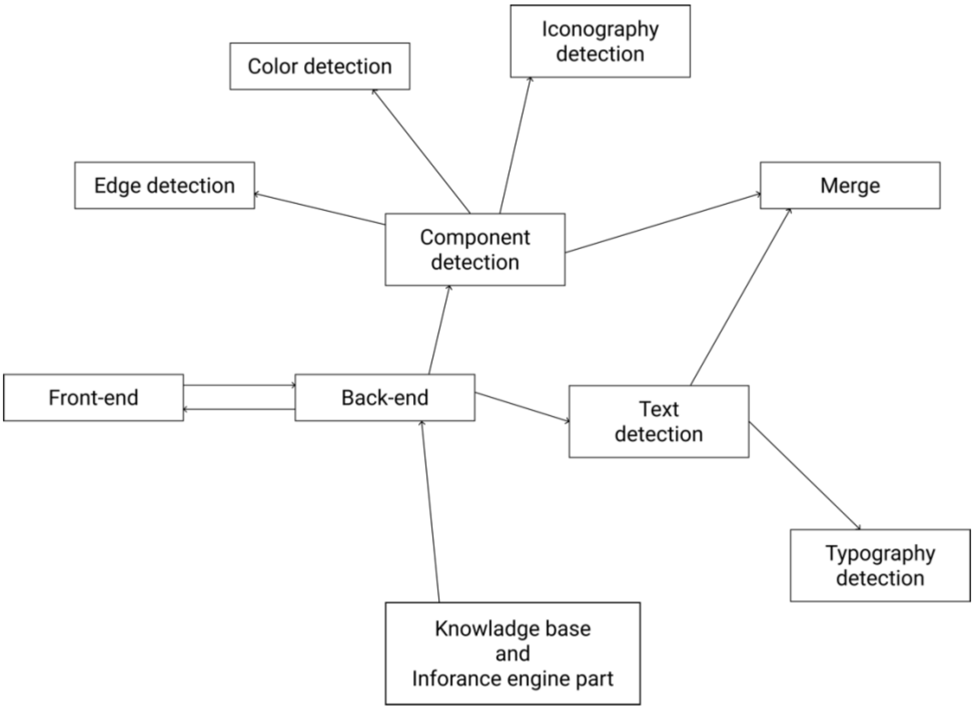


Figure 4 - Proposed system of an iOS violation detection

2.6.2 Traditional Systems

There is no traditional system to detect iOS violations. This is a research that has a domain novelty. But there are existing and traditional models for each detection model.

But the blog detection and BoK are completely a novelty to this system. There are existing models for text and component detection.

2.6.2.1 Existing Text Detection Model

According to the 2.3.1 chapter the author has explained about the previously tested models in the author process stages. They are explained along with the reasons for choosing those models. There is no existing or similar model for the violation systems but there are models with similar purposes for those single models.

(Youngmin Baek, 2019) The text detection model mentioned in this research paper and also the text detection model which is mentioned by the author in that paper are both success models. But according to this violation system, here we only need to detect the text. Character definition and boundary identification are not expected purposes from this text detection model. Because we don’t need to complex this model which was designed to aid component detection by recalling the same functions.

2.6.2.2 Existing Component Detection Model

As the first step, go to the process which converts an RGB image into a binary image.
(Ratri Dwi Atmaja, August 2016) (Tarun Kumar, September 2010)
That previously converted image comes as the input to the process of component detecting. According to the 2.3.2 chapter you have to use the OpenCV model for the detection process.
(Kari Pulli, 2012) (MeiZhang, 2017)

2.7 Review About Knowledgebase

If there is any violation with the UI which was input to the system before, the rule which was violated needs to be displayed as the output. There is a knowledge base created including these rules. But here this model hasn’t used any large/major functions. Here this model consists of functions that are created using basic python.
The rules are based on Apple HIG (Inc, 2022) And, HCI Guidelines (Google, n.d.)
Since there are no special author functions used in this model there are no research papers that were referred to while writing the functions. Here this model has used some algorithms created with basic python.

2.8 Chapter Summary

The proposed solution was first described in this chapter, this chapter consists of a description of the research that is related to components, text detection, and the research which is related to the violation detection system. There are no existing or similar systems for violation detection but in this chapter, we hope to mention the rules research and finalized resources.

CHAPTER 3: METHODOLOGY

3.1 Chapter Overview

This section presents a summary of the research domain, research challenging, implementation and development methodologies chosen to further this research.

3.2 Research Methodology

A high-quality research project should have a good domain depth, a viable goal, and effective time management throughout the research timeline. The table below contains information about the research methodology selection and reasons are there.

	Justification
Philosophy	Because research is focused on information to develop a hypothesis, and this research analyzes both qualitative and quantitative findings produced by iOS UI violation detection methods, pragmatism was chosen as the philosophy.
Approach	Here, deductive reasoning is employed since testing and assessing the concepts will show that the conclusion is correct. The proposed solution will be created and then tested to ensure it is accurate.
Strategy	The research strategy is the approach you use in order to get answers to the research questions. Interviewing domain experts will be a key technique in this study. Surveys may also be utilized with a small group of domain experts.
Choice	The research paradigm chosen will affect the research methodology used. Among the solo, multiple, and mixed techniques for selection, the mixed approach was chosen since both qualitative data such as interview results,
	document and research analysis, and quantitative data such as questionnaire data and statistical comparisons of different models will be included throughout the research.

Time Zone	In cross-sectional research, data is gathered over a period, and participants are chosen based on an observable variable. This study started with data collected from a cross-sectional design.
Techniques	Techniques such as monitoring, document reading, logical discussions, evaluation reports, interviews, and questionnaires will be used to gather data.

Table 2 - Research Methodology

3.3 Software Development Methodology

To manage the implementation phase in the SDLC, many methodologies such as waterfall, agile, and iterative prototypes are used.

Users actively engage in development using the prototype method. Thus, mistakes may be recognized early on, missing functionality can be detected, and it is able to adapt to changing requirements. Furthermore, when high-value interactions occur, the achievement of the final product is guaranteed.

The prototyping methodology was chosen from among various software development methodologies.

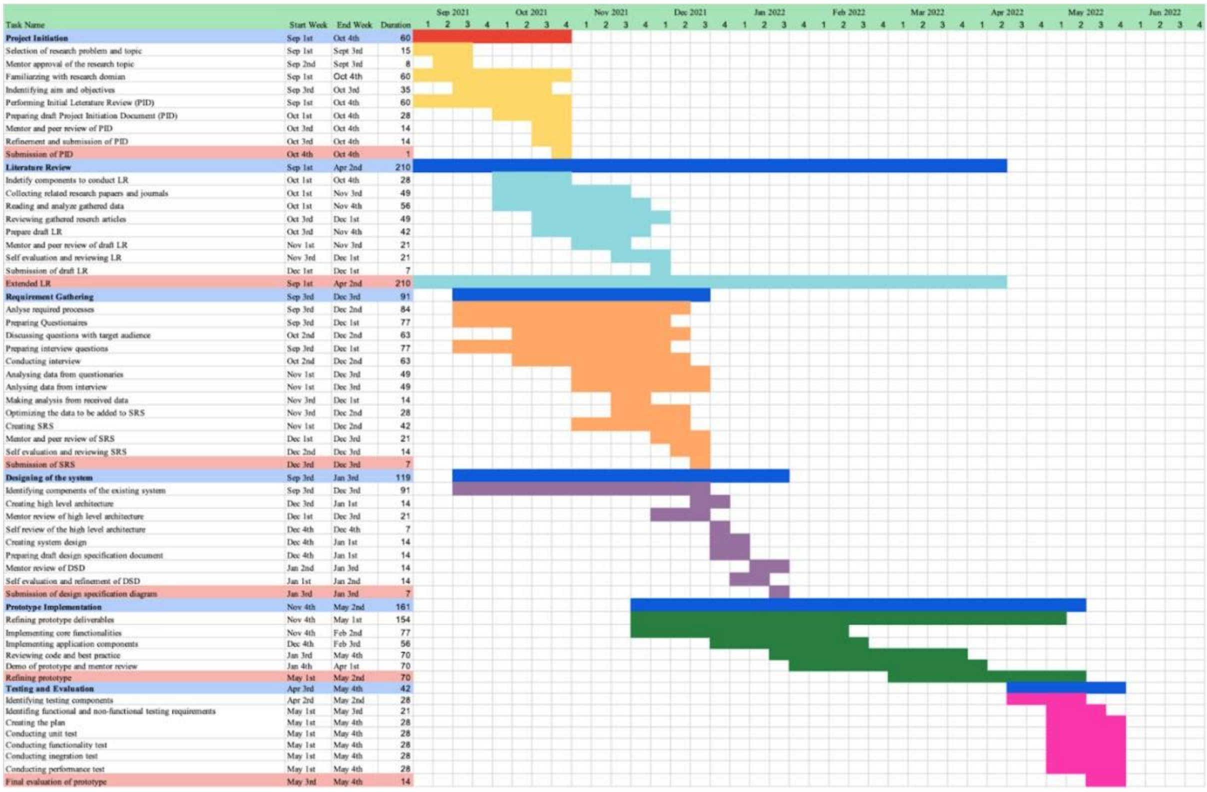
3.4 Project Management Methodology

Any project, regardless of its type, has limits such as scope, time length, and cost. These restrictions must be managed in order to provide high-quality output. Differences in requirements are possible in a research project. Thus, a project management technique is required to control and organize the project management process.

Agile PRINCE2, a hybrid of two project management methods, was chosen as the methodology.

It’s focused on business strategy reasons, but Agile is flexible and evolutionary.

3.4.1 Project Plan



Task Name	Start Week	End Week	Duration
Project Initiation	Sep 1st	Oct 4th	60
Selection of research problem and topic	Sep 1st	Sept 3rd	15
Mentor approval of the research topic	Sep 2nd	Sept 3rd	8
Familiarzing with research domian	Sep 1st	Oct 4th	60
Indentifying aim and objectives	Sep 3rd	Oct 3rd	35
Performing Initial Leterature Review (PID)	Sep 1st	Oct 4th	60
Preparing draft Project Initiation Document (PID)	Oct 1st	Oct 4th	28
Mentor and peer review of PID	Oct 3rd	Oct 4th	14
Refinement and submission of PID	Oct 3rd	Oct 4th	14
Submission of PID	Oct 4th	Oct 4th	1
Literature Review	Sep 1st	Apr 2nd	210
Indetify components to conduct LR	Oct 1st	Oct 4th	28
Collecting related research papaers and journals	Oct 1st	Nov 3rd	49
Reading and analyze gathered data	Oct 1st	Nov 4th	56
Reviewing gathered reserch articles	Oct 3rd	Dec 1st	49
Prepare draft LR	Oct 3rd	Nov 4th	42
Mentor and peer review of draft LR	Nov 1st	Nov 3rd	21
Self evaluation and reviewing LR	Nov 3rd	Dec 1st	21
Submission of draft LR	Dec 1st	Dec 1st	7
Extended LR	Sep 1st	Apr 2nd	210
Requirement Gathering	Sep 3rd	Dec 3rd	91
Anlyse required processes	Sep 3rd	Dec 2nd	84
Preparing Questionaires	Sep 3rd	Dec 1st	77
Discussing questions with target audience	Oct 2nd	Dec 2nd	63
Preparing interview questions	Sep 3rd	Dec 1st	77
Conducting interview	Oct 2nd	Dec 2nd	63
Analysing data from questionaries	Nov 1st	Dec 3rd	49
Anlysing data from interview	Nov 1st	Dec 3rd	49
Making analysis from received data	Nov 3rd	Dec 1st	14
Optimizing the data to be added to SRS	Nov 3rd	Dec 2nd	28
Creating SRS	Nov 1st	Dec 2nd	42
Mentor and peer review of SRS	Dec 1st	Dec 3rd	21
Self evaluation and reviewing SRS	Dec 2nd	Dec 3rd	14
Submission of SRS	Dec 3rd	Dec 3rd	7
Designing of the system	Sep 3rd	Jan 3rd	119
Identifying compenents of the existing system	Sep 3rd	Dec 3rd	91
Creating high level architecture	Dec 3rd	Jan 1st	14
Mentor review of high level architecture	Dec 1st	Dec 3rd	21
Self review of the high level architecture	Dec 4th	Dec 4th	7
Creating system design	Dec 4th	Jan 1st	14
Preparing draft design specification document	Dec 4th	Jan 1st	14
Mentor review of DSD	Jan 2nd	Jan 3rd	14
Self evaluation and refinement of DSD	Jan 1st	Jan 2nd	14
Submission of design specification diagram	Jan 3rd	Jan 3rd	7
Prototype Implementation	Nov 4th	May 2nd	161
Refining prototype deliverables	Nov 4th	May 1st	154
Implementing core functionalities	Nov 4th	Feb 2nd	77
Implementing application components	Dec 4th	Feb 3rd	56
Reviewing code and best practice	Jan 3rd	May 4th	70
Demo of prototype and mentor review	Jan 4th	Apr 1st	70
Refining prototype	May 1st	May 2nd	70
Testing and Evaluation	Apr 3rd	May 4th	42
Identifying testing components	Apr 2nd	May 2nd	28
Identifing functional and non-functional testing requirements	May 1st	May 3rd	21
Creating the plan	May 1st	May 4th	28
Conducting unit test	May 1st	May 4th	28
Conducting functionality test	May 1st	May 4th	28
Conducting inegration test	May 1st	May 4th	28
Conducting performance test	May 1st	May 4th	28
Final evaluation of prototype	May 3rd	May 4th	14

Figure 5 - Project Plan

3.4.2 Deliverables

Deliverables	Date
Presentation of Project Idea and Feedback from Supervisor	27th Aug 2021
Submission of Draft Project Proposal (Draft PP)	02nd Sep 2021
Evaluation of Draft Project Proposal	21st Sep 2021
Submission of Literature Review	21st Oct 2021
Submission of Project Proposal (Final PP)	4th Nov 2021
Ethics forms	4th Nov 2021
Submission of SRS	25th Nov 2021

Presentation of Implementation and Proof of Concept version and Feedback from Supervisor	17th Dec 2021
Interim Progress Report (IPR)	27th Jan 2022
Presentation of Implementation and Software Demo and Feedback	4th March 2022
Test and Evaluation Report	17th March 2022
Submission of Draft Project Reports	31st March 2022
Feedback from supervisor	10th April 2022
Thesis – The final document, presentation, and video of the research.	20th May 2022

Table 3 - Deliverables

3.4.3 Risks Associated with the Project

A project with a long-time frame is more likely to include technical risks, theoretical dangers, and even implementation issues.

Risk	**Level**	**Frequency**	**Mitigation**
Altering requirements of the project.			

As with any research, it is expected that the requirements for this research will change with each repeat of the prototype. Such alterations must be addressed.	Medium	High	As each iteration includes a brief design phase, adopting an iterative development methodology such as "Prototyping" will aid in overcoming the incidence of frequent changes to the requirements.

Lack of knowledge in relevant areas.			
Lack of understanding of the computer vision foundation and deep learning mathematical concepts.	High	High	Trying to manage time for self-study, conducting experiments, and having received expert feedback.
Fluctuations of requirements.			
Because of the nature of research, requirements can change on a constant schedule.	High	High	Prioritizing requirements, as it may be difficult to implement all of the changes in a short amount of time.
Failure to keep updated with new technologies, tools, and domain.			
Changes in the problem domain or techniques can happen all the time.	High	High	Throughout the research, look for related works, new technology updates, and have regular conversations with domain experts.
Failures in the proposed architecture.			

The proposed architecture could have faults.	High	High	Having frequent discussions with supervisors and domain experts for advice.
Unprecedented issues.			
Covid-19 pandemics, natural disasters, and political instability are likely to be an issue during the course of research.	Medium	Medium	In such situations, it is reasonable to follow the advice of the authorities and be aware of your surroundings.
Losing touch on research.			
In a quickly field like research, not keeping up with the technology and ideas can create issues.	Medium	Medium	The author must keep up with the domain and techniques. If such approach to the understanding are made, they must be analyzed and the project adjusted accordingly.

Table 4 - Risks Associated with the Project

3.5 Chapter Summary

The research methodology was already mentioned under the sub-sections of philosophy, approach, approach, choice, timeframe, and techniques. "Prototyping" was then chosen as the best software development methodology for the research. Previous to this decision being made, **Agile Prince 2** was selected as the project management methodology, with the project's deliverables as well as risk mitigation strategies discussed.

CHAPTER 4: SOFTWARE REQUIREMENTS SPECIFICATION

4.1 Chapter Overview

The main purpose of this chapter is recognizing the stakeholders. For this, you have to draw the first rich picture. After drawing the rich picture, you should gather the outline requirements of the process.

And also have to prioritize the recognized requirements and gather information. When you are drawing the rich picture, you have to identify the system and after that examine the stakeholders of its external and internal environment.

Then, in the flow diagram it describes functional and nonfunctional requirements.

4.2 Rich Picture diagram

With the rich picture, identify several stakeholders engaging with the system for various purposes.

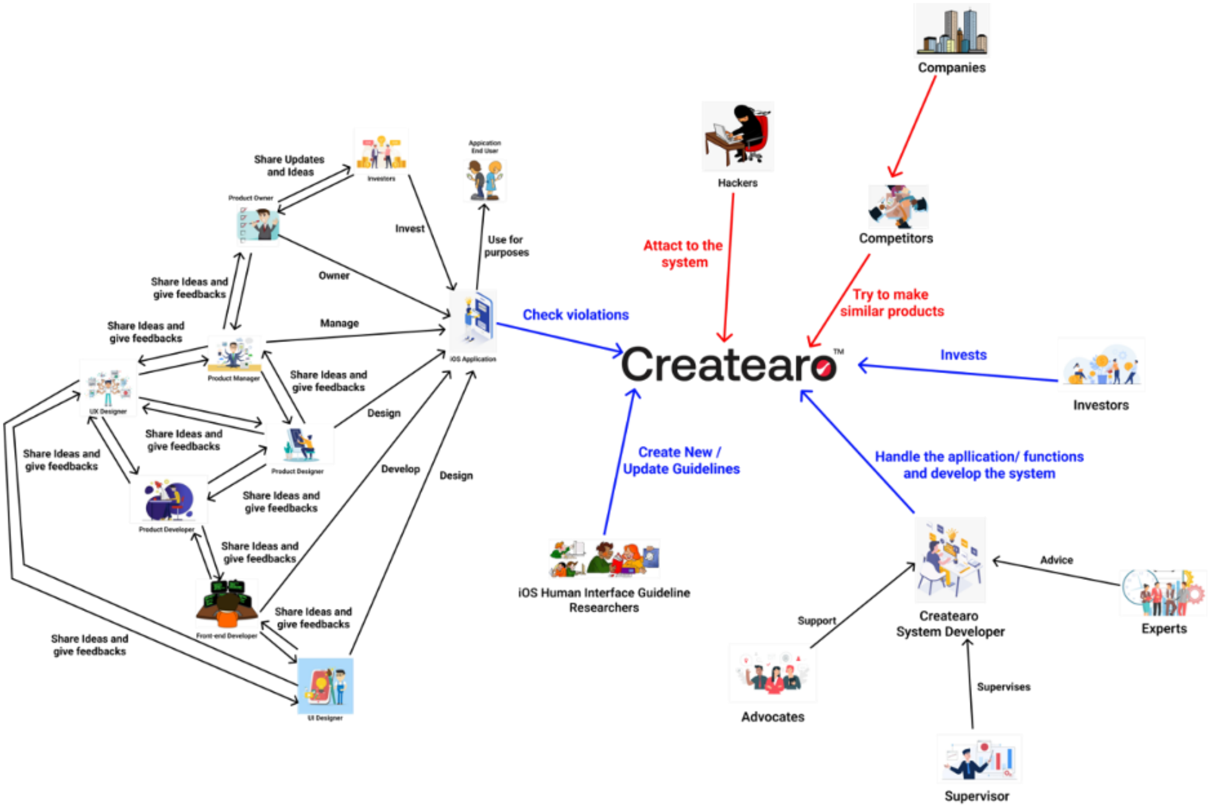


Figure 6 - Rich Picture diagram

4.3 Stakeholders Analysis

These stakeholders can use this violation model in their iOS applications. According to this end user, product manager, product owner can apply this to different domains.

The product developer will get the requirements and expert knowledge from the domain experts that will be working on the project.

The project's stakeholders are depicted as an onion model. Each stakeholder's job and point of view are clearly defined.

4.3.1 Stakeholder Onion Model

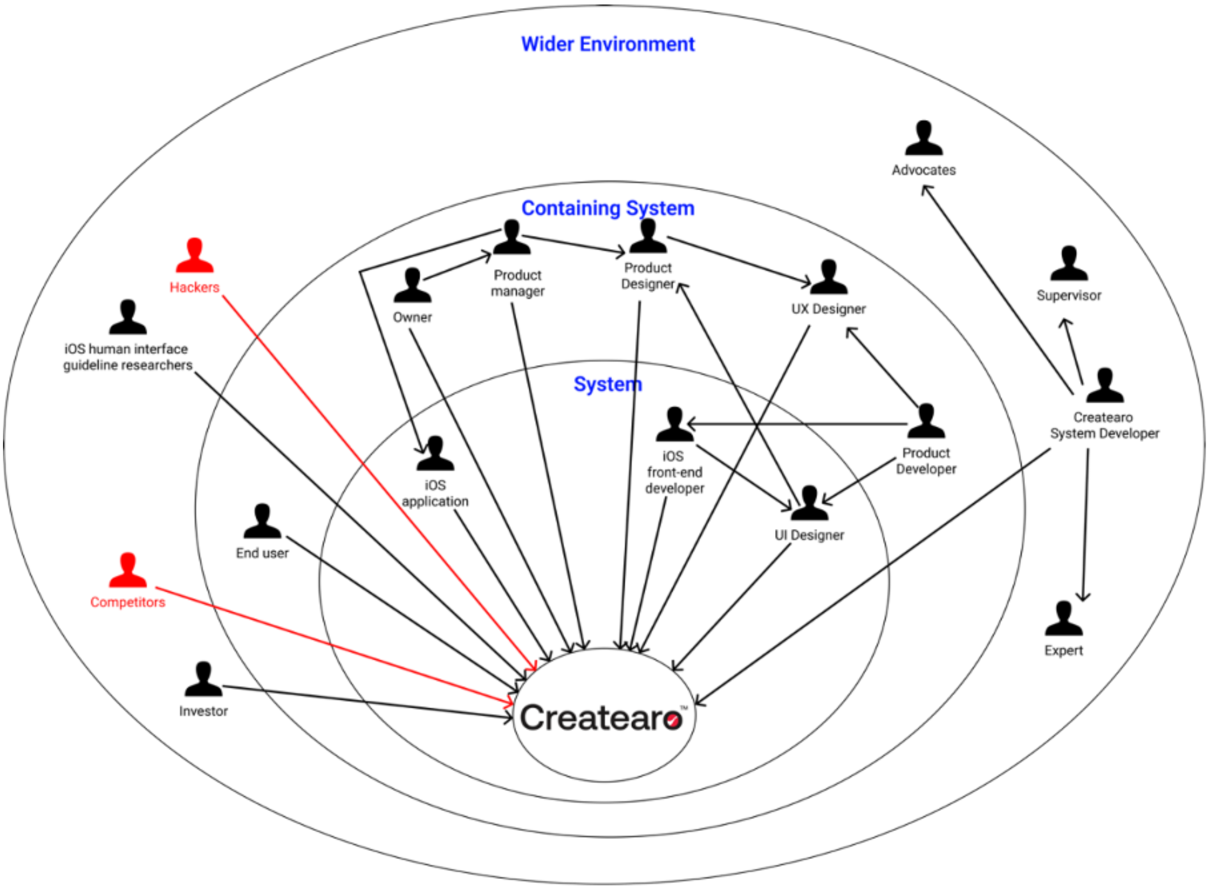


Figure 7 - Stakeholder Onion Model

4.3.2 Stakeholder Viewpoints

Stakeholder	Role	Descriptions
System Stakeholder		

iOS application	Main user	These are the main users of this system. Importance of the system is mainly affecting these people.
iOS front-end developer	The functional beneficiary	Importance of the system is mainly affecting these people.
UI designer	Management of the product's maintenance and enhancement	Importance of the system is mainly affecting these people.

Containing System Stakeholder		
UX designer	The operation beneficiary	These are the main users of this system. Importance of the system is mainly affecting these people. And also, these are the base of interviews and questionnaires.
Project owner/ manager/ developer	The operation beneficiary	People that get benefits from the result of the system.
Application end user	User	They can use non-violated product with best UI.
Wider Environment Stakeholders		
Expert	Expert	It will be evaluated by a team of specialists to see if it reaches the required standards and if it can be improved upon.

Investor	The financial beneficiary	Investors will put money into the product's development in the hopes that it will return a profit.
Hackers	Negative stakeholders	Hackers may attempt to modify the system's important input images in some other way, or they may just try to create damage to the system.
Supervisor	Supervises	Provides the necessary guidance and support to ensure the effective completion of a product.
Advocates	Support	Provides the necessary guidance and support to ensure the effective completion of a product.
CREATEARO System Developer	Develop	Develop the violation detection system. Handle the application/ functions in the system.
Competitors	Negative stakeholders	Try to develop similar products.
iOS human interface guideline researchers	Domain Expert	Develop new rules or update rules.

Table 5 - Stakeholder Viewpoints

4.4 Selection of Requirement Elicitation Method

Requirement elicitation entails the use of a variety of tools and methodologies to collect information on the project's requirements. In this part, a variety of such possibilities are examined, and the reasons for their selection are explained.

Prototyping, literature review, interviews, and questionnaires are covered.

4.4.1 Justifications for the Selection of RE Techniques

1. Literature Review

The insights and knowledge gained from the literature review might be regarded the first stage in the requirement engineering process. Gaps and difficulties in current systems are easily recognized by thorough research of the literature, and these gaps are extremely useful for developing engineering requirements.

As a result, a detailed examination of the study topic, current systems, and potential methodologies and technologies was done.

2. Formal Interview

In general, there are two sorts of surveys that may be used to gather requirements. **Formal interviews**, on the other hand, are conducted using **questionnaires**.

The project necessitated conducting official interviews with specialists, thus they were chosen from the two options. Because the scope of the project has been confined to a certain area and a defined set of methodologies and technologies, this choice has been made. The decision was made to consult with seasoned domain and technology experts to acquire their take on the matter. When it comes time to finalize the prototype's scope, what they have to say about the gaps and the project's overall scope will be an invaluable resource for developing project requirements.

In addition, their knowledge would allow them to perceive a variety of possible solutions to the project's challenges. As a result of these considerations, this was decided to perform formal interviews with specialists in order to gain their input.

3. Questionnaire

Using participants as just a sample from of the broader public or society, the author distributed a questionnaire among system users to gather needs. A questionnaire like this will assist the author get a better sense of what people are thinking and what they anticipate the prototype to accomplish.

In addition, a questionnaire like this can assist determine whether the product will be useful or whether it can meet its stated goals.

4. Prototyping
The project's prototype has a simple and direct goal. The goal was to create an easy to-use user interface that would streamline the process of publishing an app's screen and related meta data.

Table 6 - Justifications for the Selection of RE Techniques

4.4.2 Justifications for the non - Selection of RE Techniques

1. Requirement Workshops
The ability to bring together a diverse collection of stakeholders for a requirement workshop is a major drawback.
2. Brainstorming
A broad group of stakeholders is required to facilitate brainstorming sessions. For the facilitator and scribe duties, two individuals are forced. Due to a limited time schedules, stakeholders could not be included in this project.

Table 7 - Justifications for the non - Selection of RE Techniques

4.5 Discussion of Results

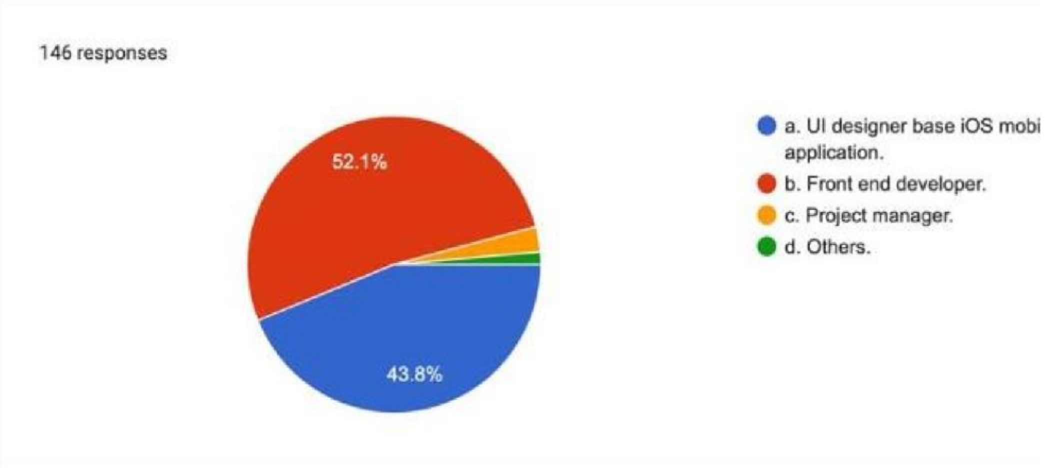
4.5.1 Findings from LR

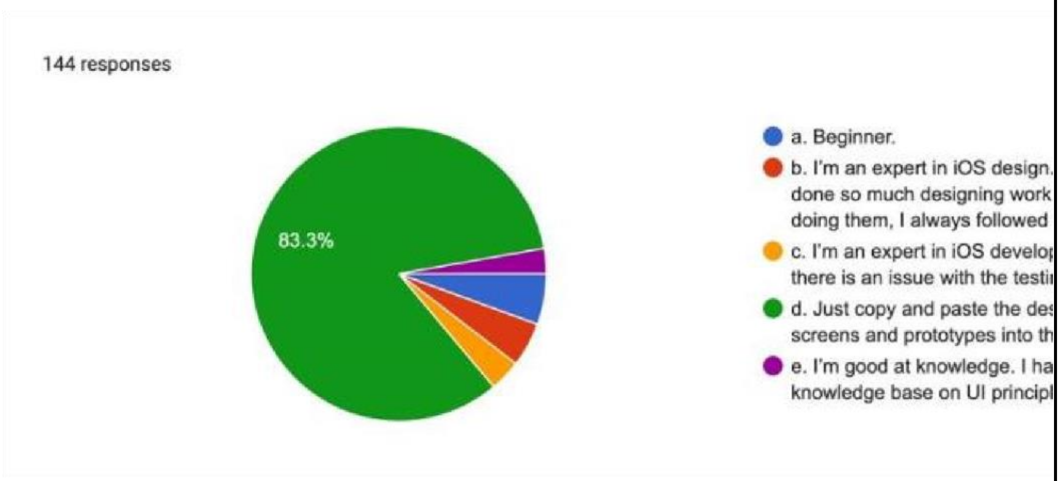
Findings	Component and text detect interpreted.
	When it comes to the design of user interfaces, one of the most common techniques for detecting elements is the use of computer vision (CV). For example, a mobile app image might be used as an input for UI Element Detection (UIED), as well as UI designs made in Photoshop or Sketch. Once the text and graphic UI components are detected and classified, they are exported as JSON files for future use. UI text and visual components, such as buttons, images, and input bars, are detected by UIED using a two-part architecture. Detection of text is handled using Google OCR. To find and classify graphical components, it makes use of traditional CV techniques and a CNN classifier.

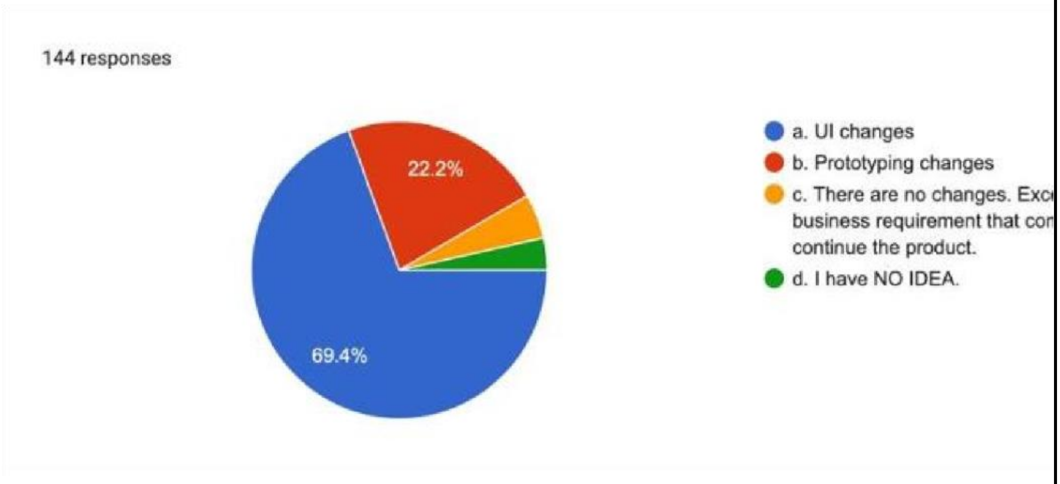
Citation	(Mulong Xie, 2020) (MulongXie, 2021) (Valéria Lelli, June,2016)

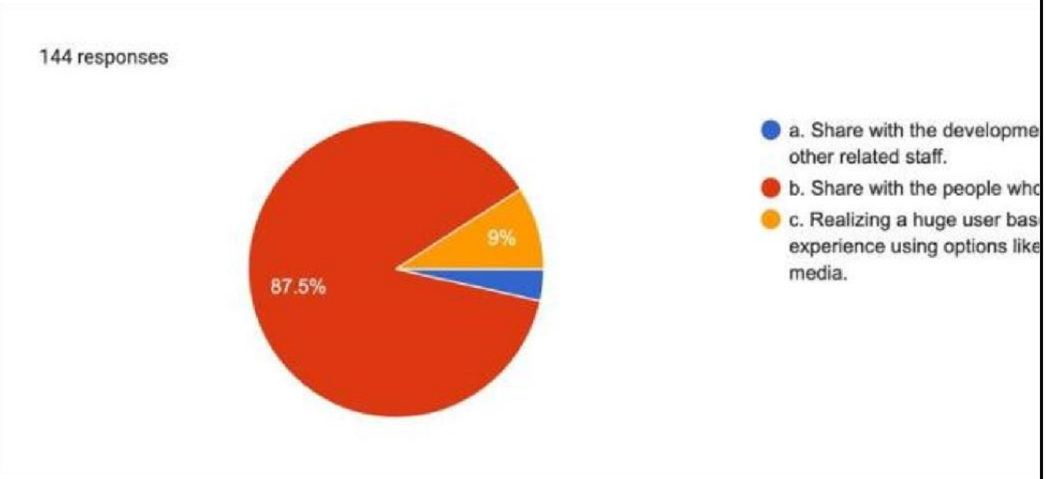
Table 8 - Findings from LR

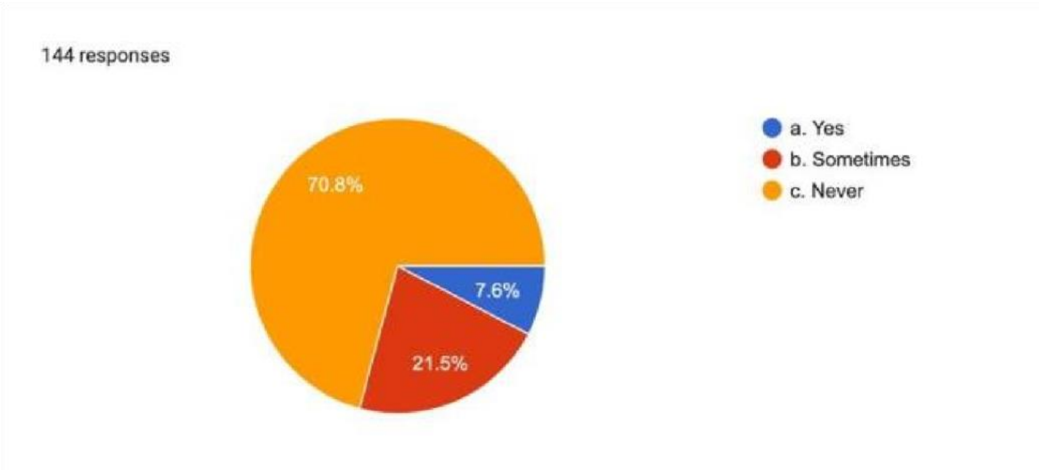
4.5.2 Questions

Question – no.1	Choose your passionate carrier, a) UI designer base iOS mobile application. b)Front end developer. c) Project manager. d) Others.
Aim	User identification and filtering.
Findings	 As it was expected, almost all the participants (more than 95%) of the survey will be a front-end developer or a UI designer. From here we can associate that the people who were involved in this survey have knowledge about this. To ensure that replies came from participants with sufficient experience, the survey was only extended to those who replied, "UI designer base iOS mobile application or front-end developer."

Question – no.2	How would you describe yourself in the domain of UI principles? a) Beginner. b) I’m an expert in iOS design. I have done so much designing work. When doing them, I always followed the guidelines. c) I’m an expert in iOS development. If there is an issue with the testing, I have the knowledge to solve it also. d) Just copy and paste the design screens and prototypes into the code. The principle doesn't bother me. e) I’m good at knowledge. I have a good knowledge base on UI principles. Because I'm doing research and paying attention to changing principles. But work experience is limited.
Aim	To get an idea about what kind of knowledge that the survey users have about principles.
Findings	 There 5.6% at the beginner level and 4.9% from the second stage. 3.5% of developers have knowledge about iOS design principles as well. 83.3% is on the

	fourth stage. Any application that hasn't been rejected will come out with bad UI and UX from the app store because they don't have knowledge about violation. Then the other people on 4. They are persons that stand as a UI consult base. They have up to date knowledge about principles and trends.
Question – no.3	After you have designed all the screens or done the development of the application and then what kind of changes can you see after you submit the final output in the app store, a) UI changes b) Prototyping changes c) There are no changes. Except for a business requirement that comes to continue the product. d) I have NO IDEA.
Aim	To identify what kind of changes are coming. You have to find out that there is a user base for this violation detection tool.
Findings	 Many have received UI changes. Prototyping changes are not at all. Requirements have come to continue too. If there are UI changes that means the designer and developer has spent their time to re-work the same task again and again.

Question – no.4	How to do the user testing of MVP that comes after the UI design? a) Share with the development team and other related staff. b) Share with the people who we know c) Realizing a huge user base to share experience using options like social media.
Aim	To identify if there was a chance to detect the violation.
Findings	 There is a higher probability to detect the violation if it is sent as an MVP to a large database. Cost will be less effective for the re-works because of the use of like this tool in the development level.
Question – no.5	Do you always refer to iOS principals with iOS design or development? a) Yes b) Sometimes c) Never
Aim	To recognize that how many are there of doing design and development according to the UI principles.

Findings	 There are two sections of violation. One is a UI that doesn't have responsiveness and the other one that doesn't care about the UI principles. From this survey we can identify that most of the user base don't care about the principles.
Question – no.6	How successful would it be if given the chance to detect violation by inputting an image and metadata file/ hierarchy json file (optional)on the application screen? 1- Very Poor. 5- Excellent.
Aim	Research into establishing whether professionals in the relevant field are examining the suggested solution is important.

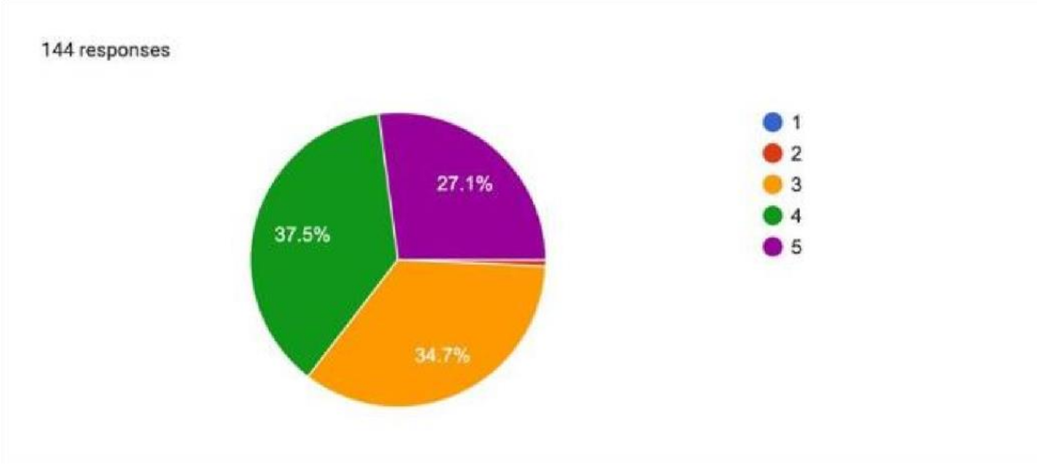
Findings	 37.5 % of those polled thought the approach was good in every way. 27.1 % found this study to be excellent. 34.7 % of those polled said they were satisfied.

Table 9 - Questions

4.5.3 Prototyping

To further test the suggested architecture and gather and validate requirements, we used the prototyping technique in collaboration with domain and technical experts. According to Inductive Thematic analysis, the following table shows all of the findings.

Codes	Theme
The architecture seemed to be reasonable, well-planned, and developed; the executed accordingly to be on the correct track.	Design Validity
Alternate setups for hyperparameters.	Configuration
Domain-specific.	Applied Domain

Theme	Conclusion	Evidence

To begin - Rich picture prototype feature diagram.		
Design validity	The recommended architecture and system controller is designed based on the prototype have been accepted by all of the experts.	"The structure appears to be well thought-out." 'Well thought out and developed to provide a more effective solution.' "Aesthetics and layout that adhere to industry standards." “There seemed to be a solid path to take for improving the quality of a final picture."
In middle - Core components		
Configurations	In the opinion of the most of experts, – the opportunity to configure hyper - parameters using the GUI is a gets a lot.	Developers would if they could change hyperparameter setups from their perspective. It would be nice to get a customized situation where the hyperparameters may be adjusted using the GUI, rather than the default scenario.
Applied domain.	Some experts have suggested that iOS violation detection should only be applied to a specific domain at a moment.	In order to see how well the model will perform with various hyperparameter values, "apply for a given domain." It would better. If the modules can be connected with other general domains.

Table 10 - Prototyping

4.5.4 Self-evaluation

The research benefited from self-evaluation at every stage, especially when it came to analyzing the scope, making important decisions about the many directions the study may go, and setting the discovered needs in order of importance in relation to the schedule.

4.6 Summery of Findings

Findings	LR	Formal Interviews	Survey	Self - evaluations	Prototyping
Verify the existence of a previously unexplored research area.	✓	✓	✓		
The proposed method's viability.	✓	✓	✓		✓
Prototype system must accept any application screen as input and produce the image that is presented by violations.	✓	✓	✓		
It should be possible to display components and text in the same json file at the detection part.	✓			✓	
A graphical interface is a must for a prototype.		✓	✓	✓	
It should highlight the place where is the violation of the output report has placed.				✓	
The user interface must be straightforward and easy to learn, making it more convenient for everyone.			✓		

Optimize resources by using architectures that can be easily transferred.		✓			

Table 11 - Summery of Findings

4.7 Context Diagram

Before implementing a system, it is important to define its boundaries and identify its internal and external components. The system's context is depicted in the figure below.

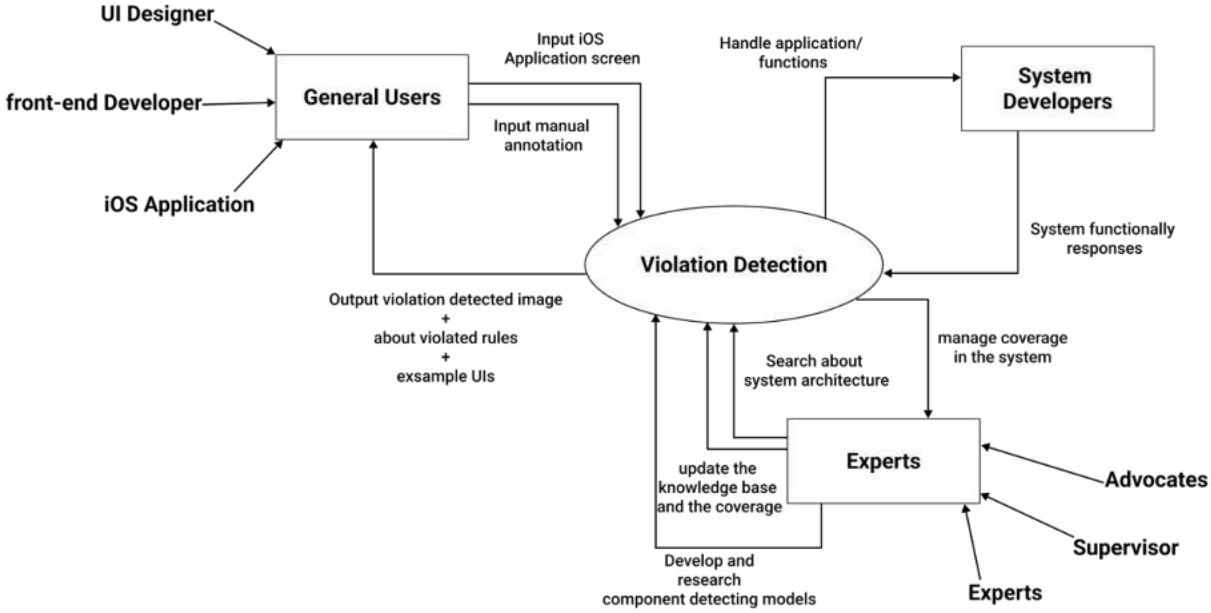

UI Designer
front-end Developer
iOS Application
General Users
Input iOS Application screen
Input manual annotation
Handle application/ functions
System Developers
System functionally responses
Violation Detection
Output violation detected image + about violated rules + exsample UIs
manage coverage in the system
Search about system architecture
Experts
update the knowledge base and the coverage
Develop and research component detecting models
Advocates
Supervisor
Experts

Figure 8 - Context Diagram

4.8 Use Case Diagram

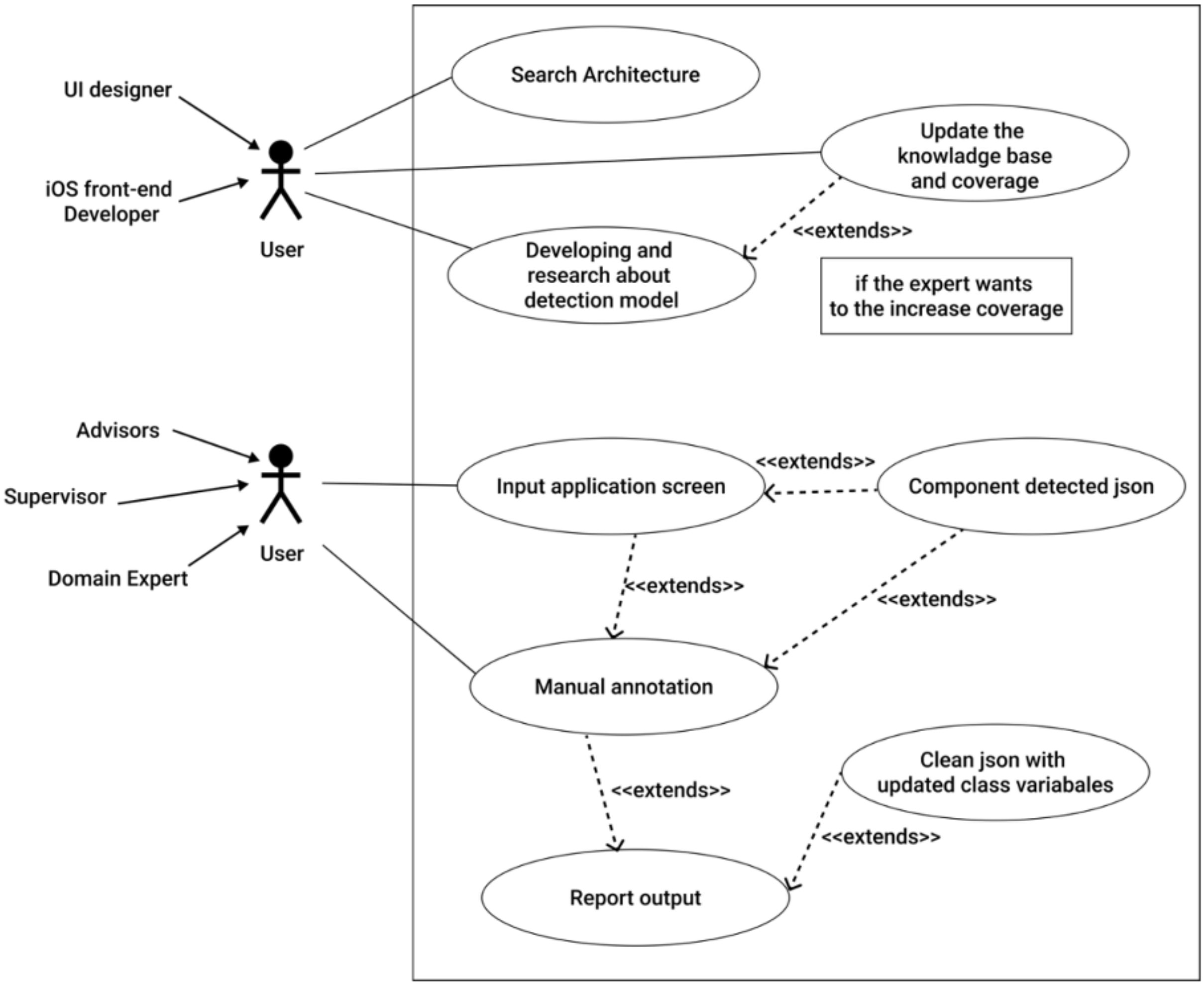


Figure 9 - Use Case Diagram

4.9 User Description

Table 12 - User Description - Use Case Name - no.1

Use Case Name – no.1	**Input application screen**
Description	Users can upload a picture of their application to see if it has been violated or not.
Participating Actors	To identify infractions, the user must possess a picture.
Pre-conditions	Display an Error Message and an uploaded Image.

Included use cases	None
Main flow	The application screen to be uploaded is chosen by the user. The application screen is attached if it is in the right format.
	An error notice is given if the image format is incorrect or if it is not an image. Principles will be violated if the user continues with the selected screen.
Alternative flow	None
Exceptional flow	None
Post conditions	The application screen that has input should detect the violation and highlight it.

Use Case Name – no.2	**Manual annotation**
Description	Users can annotate components in the detected screen
Participating Actors	User
Pre-conditions	Should select the components that the user wants to detect from the components that have been detected.
Extended use cases	If the user wants, you can select the component. If not, the system will identify all the violations of the component that have been detected.
Included use cases	None
Main flow	User should select the components that he wants.

Alternative flow	None
Exceptional flow	None
Post conditions	Can be made a clean json as mentioned in the logic tier.

Table 13 - User Description - Use Case Name - no.2

4.10 Requirements

The MoSCoW approach was used to rank the relevance of system needs and assign them a priority level.

Table 14 - Requirements (MoSCoW)

Priority Level	Description
Must have (M)	In order for the prototype to work properly, a functional need of this level must be implemented.
Should have (S)	Even if they aren't essential to the projected prototype, important criteria have a big impact.
Could have (C)	The project scope does not include any needs that are desirable.
Will not have (W)	It's not a necessity at this stage to deal with the system's weaknesses.

4.10.1 Non- Functional Requirements need to be prioritized using MoSCoW principles.

Table 15 - Non- Functional Requirements need to be prioritized using MoSCoW principles.

ID	Requirement	Description	

1	Performance	Considering that an input from the user an application screen to the system, the first identification procedure and human annotation and final output must not take too much time. During the processing of the picture, it is necessary to ensure that the program does not crash.	M
2	Output Quality	It is imperative that the quality of the output image match the quality of the input screen. The primary objective of the project is to provide the user with high-quality output.	M
3	Security	A greater degree of security is necessary since the system processes private photographs being sent in by all the user and needs to provide better security.	M
4	Usability	A solid user experience is required for the prototype.	C
		For the first time user, the application should have a low learning curve, making it easy for them to get up and running quickly. To keep people informed during the entire process, it is essential to keep them up to date.	
5	Scalability	The application's burden is likely to increase over time when the prototype is implemented. When the number of users increases, the system should be able to deal with the increased workload efficiently.	C

4.10.2 Functional Requirements need to be prioritized using MoSCoW principles.

ID	Requirement Description	Use case mapping	
1	Selecting and uploading in the application screen that users wish to detect screen violations must be possible.	Input application screen	M

2	The frontend must validate the application screen file type.	Input application screen	M
3	The model must be able to recognize the different components in the screen using a pure computer vision base.	Component detection	M
4	The model must be able to recognize the text in the screen using Tesseract OCR.	Component detection	M
5	Input only the components that the user wants to detect the violation from the components that have been detected by the previous flow.	Manual annotation	S
6	Users can increase the detection coverage with the screen that json file has uploaded.	Input application screen	S
7	There has been created a clean json relevant to the inputted components, to input for the final detection.	Clean data file with class update	M

Table 16 - Functional Requirements need to be prioritized using MoSCoW principles.

4.11 Chapter Summary

This chapter explains who the project's stakeholders are and what their role is in the overall success of the project. The approaches for eliciting needs were then identified, and the findings, both quantitative and qualitative, were analyzed.

It was decided to conduct a thematic analysis. Functional and non-functional requirements were established as a result of the study's findings It was in this chapter that a use case graphic and its related text were developed.

CHAPTER 5: SOCIAL, LEGAL, ETHICAL, AND PROFESSIONAL ISSUES

5.1 Chapter overview

This chapter summarizes the social, legal, ethical, and professional issues and suggests solutions.

5.2 SLEP Issues and Mitigation

Table 17 - SLEP Issues and Mitigation

Social
1. In terms of violating the privacy, the names of the evaluator or interviewees were not mentioned in the thesis. 2. The questionnaire did not collect any user data and was fully anonymous. Users were informed that the survey was anonymous inside its title and that by completing it, they consent to the author using the collected data. 3. As it is, distributed survey responses were never used in the project thesis. Only the quantification was added, but the privacy, security, and identification of the collected data were respected and protected. 4. This research contains no ethical, political, or religious appropriations.
Legal
1. Open-source licenses are used to license specific tools, languages, and libraries. (Python, Keras, TensorFlow, OpenCV and etc.) 2. Copyright settings for publications connected to the research were properly followed and transmitted to publishers.
3. Interviews were only recorded with the permission of the interviewees. If they rejected to record the interview, key points were taken down and will be created in the form of transcriptions.
Ethical

1. Plagiarism is heavily monitored, and there is no plagiarized text in the thesis. The literature is accurately cited and referenced. 2. Participants who completed the surveys were made aware of the project through the questionnaire description, as well as how they are contribute to the project by completing this questionnaire.
Professional
1. During the design and implementation of the project, industry standards and best practices in software engineering will be followed. 2. The whole prototype was created in properly secured settings that have been password protected and maintained up to date with the most recent security updates. 3. All project limitations were fully described in the research thesis, and such limitations were also revealed to evaluators during interviews.

5.3 Chapter Summary

This chapter's only purpose was to address any social, legal, ethical, and professional issues raised by the project, as well as how they were mitigated and handled. All of the main SLEP concerns were addressed at various points.

CHAPTER 6 : SYSTEM ARCHITECTURE AND DESIGN

6.1 Chapter Overview

This chapter goes into great detail about the project's design. It includes everything from the system's core components to its user interface. Research, interviews, questionnaires and prototyping results were all used to make design decisions for this project. The reasoning for various design choices is also explored in this section.

6.2 Design Goals

Before making any design-related decisions, it is necessary to evaluate the design goals.

Design Goal	Description

Ease to use	The system must be structured in such a way that the end user can use it simply. It must be a user-friendly and simple-to-learn system. The system has a guideline to describe how to use this violation detection. Additionally, from a developer's perspective, it must be an easy-to-learn factor that promotes new developers to join the development process in order to improve existing system and add new features.
Performance and Reduce Latency	Python is used for the back end. Django is also a Python framework. Because of this, latency is reducing. A system's performance also has a lot to do with how well it works. Errors, lags, or other problems must not happen at either the logics or in front of the process. This is very important.
Reusability	It is important for the modules of the system to be created in a way that allows them to be changed. This is especially important for research that helps other researchers in the field. If anyone can change a part of the system with another, they can try some new things or improve the system. Example they can add new components/ design aspects/ design dimensions according to the future trends.
Correctness	Components are detected messy at the first detection. So that there is an option to select the components that the user wants through a manual annotation.
Scalability	The system should be able to use large datasets for training because there will be more datasets for iOS apps in the future. Also, it should be able to handle different kinds of new loss functions that are added, because that will be a big factor in how well the system works. Also, the system must be able to run on a server that can serve a lot of people at the same time.

Table 18 - Design Goals

6.3 System Architecture Design

System architecture design is very important to how well the system can meet its design goals. So, the right design architecture should be used.

An analysis is available that compares the architectures that were looked at and the reasons why they were chosen.

6.3.1 Tiered Architecture

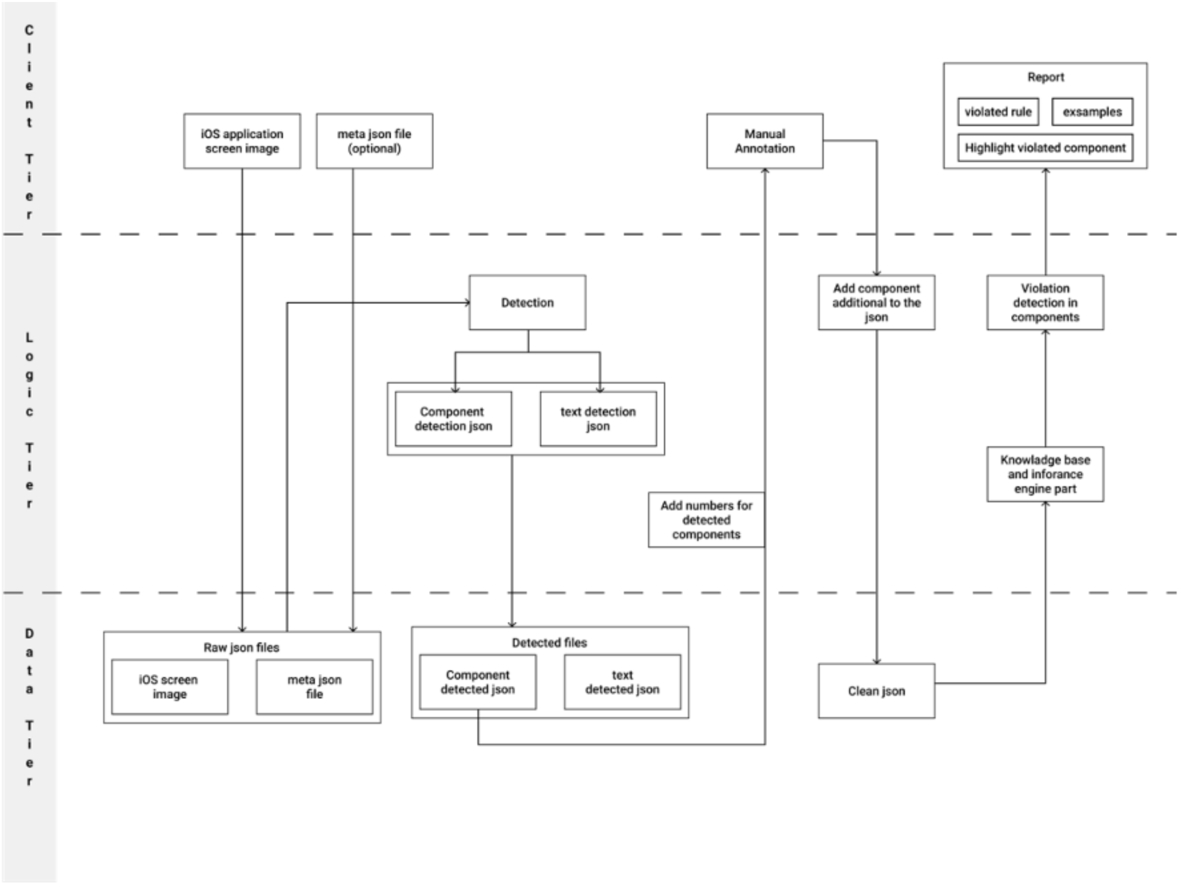


Figure 10 - Tiered Architecture

It is possible to gain a number of advantages by employing n-tier architecture. **Security, flexibility, and scalability** make up this list.

Simple management: Each layer may be added to or modified separately, without impacting the other levels in any way.

However, using **Layered architecture, scalability is difficult since the framework's structure does not allow for expansion.**

They can be a pain to keep clean. Since each layer receives the data from the one above it, there is dependency between them. It is not possible to process data in parallel.

As a result, a three(n)-tier architecture was proposed.

Client (Presentation) Tier, Logic Tier, and Data Tier comprise the contributing system's tiered architecture. Connects the Data Tier, which contains all of an application's information, to its Client Tier, which displays it to the user. When it comes to operations, the Logic Tier is simpler and more flexible.

Tier	**Description**
Presentation tier (Client tier)	
Image File	The place where input the application screen that the user wants to see the violations.
Meta json file / Hierarchy json file	Place that user inputs the json file. But this one is optional.
Manual annotation	The image that comes after the first-time detection of components from the logic tier is sometimes more messy. Sometimes non components are also detected as components from this method because it is based on computer vision. Considering this fact, the image that has come after the first detection should send for the interface again. There can be annotated components that the user wants manually.
Report	This is the final output. It comes from e categories. One is highlighting the place that violation has placed. Second one is showing the non-violation samples according to the violated components. Conveying the violated rule of the screen or the input component is the last one.
Logic tier	

Detection	Here two main detections are happening. 1. Text detection- Has used Tesseract OCR. 2. Component detection- pure computer vision base technique. Here the detection is based on the screen that the user has input at the client tier.

	But we can't detect an inner text with components because of the algorithm and methods that have been used. Text and component detections are coming as separately. But in a further research part, they have found a model that can get these two (text and components) into the same json file. Definitely hope to replace it on the final submission. Add numbers for detection component- because of the messiness about the detection part of the current model, there is a manual annotation path of the flow as mentioned in the client tie. Adding a numbering for all the components that has come from the 1st erection is the thing that is happening in this part. Because it is user friendly for the user in the manual annotation. Update class in jason for detection part- updating the class according to the ID of the jason what is coming from the component detection with the inputted data in the manual annotation.
Knowledge base and Inference engine part	Checking about what kind of components are there in the input screen from the user annotation result.
Violation detection in component	After detection, then called by the functions of that components.
Data tier	

Raw files	Files that come from the user. You can upload only the image file. Otherwise, you can add a json file with the image file. But a meta json file is optional. Also, there is an advantage for the user by adding a metadata file. So that increased the score of guidelines that can be covered from the model.
Detected json file	The file that was created before the detection of manual annotation.
Clear json	The jason, that is made after removing the other components while remaining the components which the user needs to detect the violation of the image. Here, there is a clean json that can be updated the class and transfer to the knowledge base as mentioned in the logic tier.

Table 19 - Tiered Architecture

6.4 System Design

Justifications and diagrams relating to system design are included in the system design process.

6.4.1 Choice of the Design Paradigm

In the course of software development, the choice of a design paradigm or approach is critical. The approach to software development is dependent on a variety of factors, including the environment in which it is being developed, the needs of the end user, the type of software being developed, and the amount of time available.

Two design paradigms stood out among the several that were explored for the project:

1. Object Oriented Analysis and Design (OOAD)
2. Structured Systems Analysis and Design Method (SSADM)

This system falls within the category of generative modeling, where the whole paradigm is nondeterministic, so the **Structured Systems Analysis and Design Method (SSADM)** was used.

In order to build it, it's made up of a series of modules, each with a particular set of duties assigned to it. That is why object-oriented principles mapping will be difficult and won't provide developers the freedom they need when working on a project of this kind.

6.5 Design Daigram

6.5.1 Data flow diagram

The Dataflow Diagram is composed of many components of the research contribution, one of which is the violation detection component. These modules were in charge of certain tasks, and the figure depicts the data flow between them.

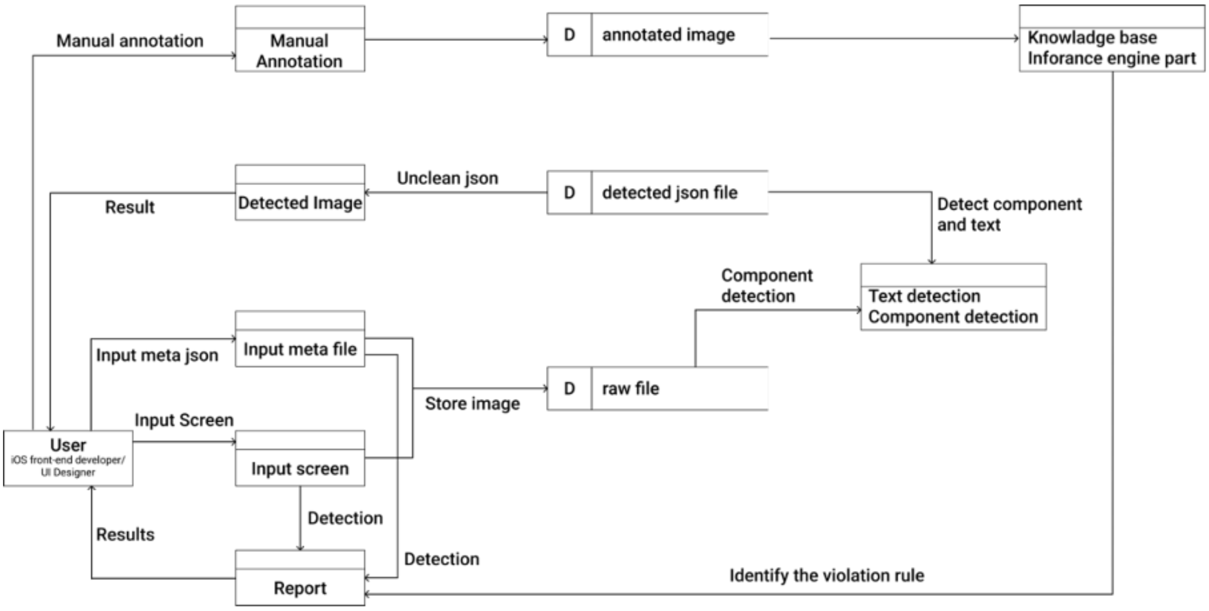


Figure 11 - Data flow diagram

6.5.2 Sequence diagram

The sequence diagram shows the five primary entities that make it up the violation detection system, as well as the interactions between the user and the system's core entities.

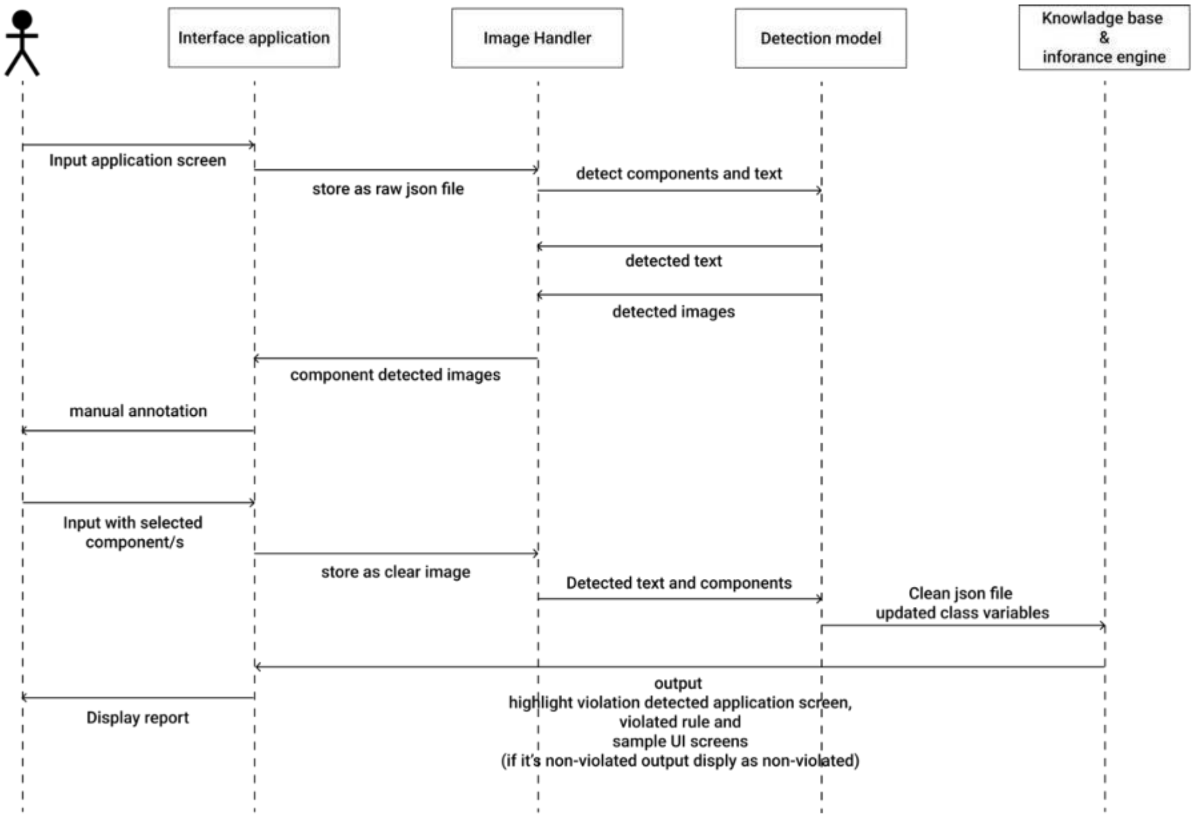


Figure 12 - Sequence diagram

6.5.3 Algorithm Design

There are two sections in the detection part.
One is the text detection part and the other one is the component detection part.

Here for the text detection part, use Tesseract OCR what a model that has already exists. Computer version-based models have been used for component detection. There is not a use of training models in this whole process.

6.5.4 UI Design

The project's UI (User Interface) has a simple and direct goal. The goal was to create an easy to-use user interface that would streamline the process of publishing an app's screen and related meta data.

6.5.5 User Experience flow diagram

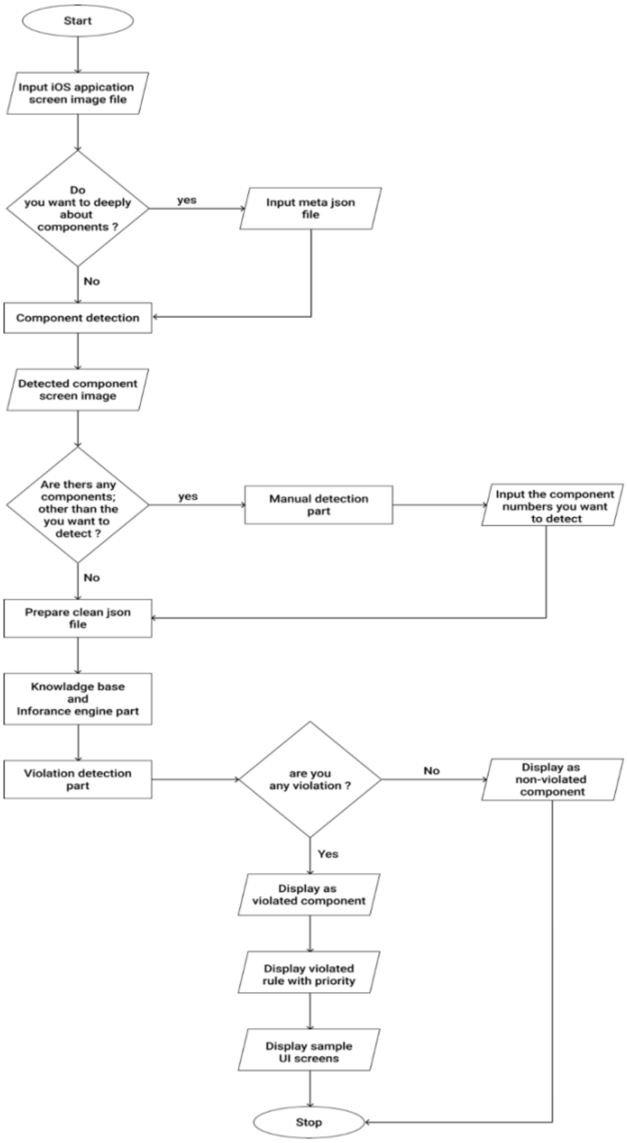

Start
Input iOS appication screen image file
Do you want to deeply about components ?
yes
Input meta json file
No
Component detection
Detected component screen image
Are thers any components; other than the you want to detect ?
yes
Manual detection part
Input the component numbers you want to detect
No
Prepare clean json file
Knowladge base and Inforance engine part
Violation detection part
are you any violation ?
No
Display as non-violated component
Yes
Display as violated component
Display violated rule with priority
Display sample UI screens
Stop

Figure 13 - User Experience flow diagram

6.5.6 System Process Flow Chart

When a user uploads their application screen and meta data file to the system, the system process flow chart shows how the system works and how its choices are made.

This mostly demonstrates how the application's logic tier is structured and regulated, and how these choices and processes are implemented.

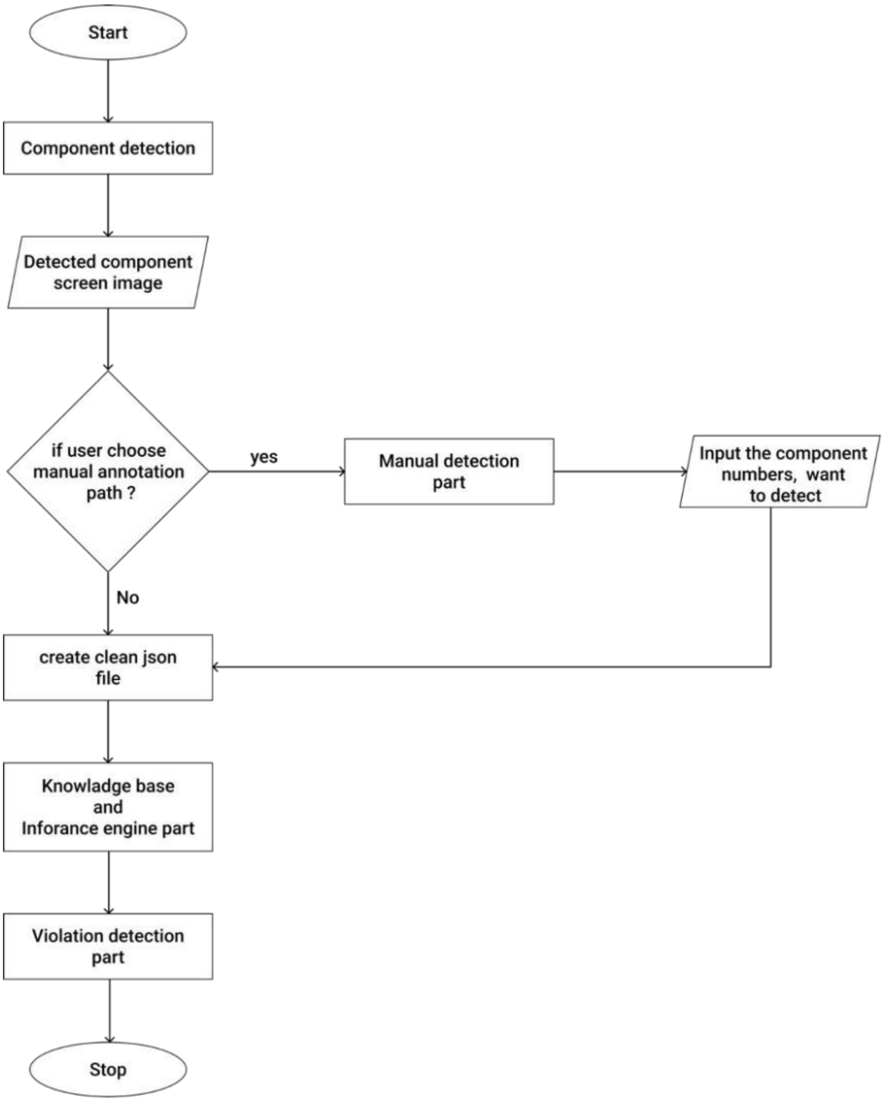

Start
Component detection
Detected component screen image
if user choose manual annotation path ?
yes
Manual detection part
Input the component numbers, want to detect
No
create clean json file
Knowladge base and Inforance engine part
Violation detection part
Stop

Figure 14 - System Process Flow Chart

6.6 Chapter Summery

Using findings from a literature analysis and requirements collected through interviews, surveys, and prototyping, this chapter explains in detail all of the architectural decisions that have been made.

The data-flow diagram modularized the whole system into separate components and depicted the flow of data between those components, while the tiering architecture provided an overview of the three primary tiers.

There were five primary entities that were shown in a high-level sequence diagram that indicates how the system would respond to human interaction.

The main notion of the system's intended user interfaces was demonstrated in the UI design. A flowchart of the system's primary choices and processes, dependent on user interactions, was provided as an explanation.

CHAPTER 7: IMPLEMENTATION

7.1 Chapter Overview

This chapter will give an overview as to how the system was implemented and the decisions that were made throughout the implementation. The outcomes of the literature research, requirements engineering, and system design, as well as the architecture, all influence the decisions and methods of implementation.

An executable version of the concept would be created.

7.2 Technology Selection

7.2.1 Technology stack

The following technologies were selected to be used throughout different parts of the project. Python has been used as the main programming language. NumPy and pandas are used as pythonic frameworks. In the front end, we used react.js. There, we used a material UI framework because we needed to make a rich UI.

Over that author used a computer based OpenCV for the component detection part. For the text detection part, Tesseract OCR has been used.

Git has been used to control the version. As IDES, PyCharm and Visual Studio Code has been used.

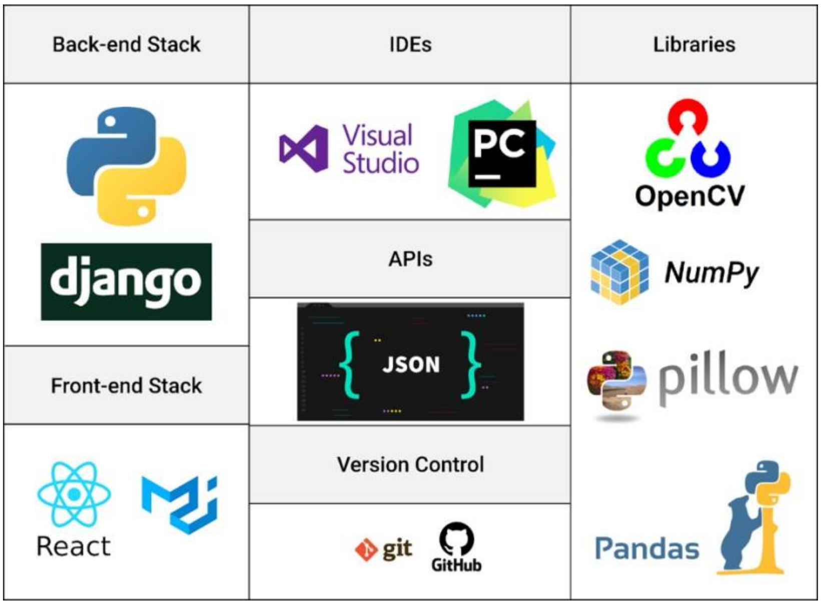


Figure 15 - Technology stack

7.2.2 Data Selection

Model is created by the overall deep learning and machine learning. But the end-to-end processes of this model haven't used a training base process anywhere. In the text detection part, there has been taken a Tesseract OCR algorithm that has already existed.
Pure computer-based techniques are used for component detection.

7.2.3 Development Framework

Because of the computer-based models, used the OpenCV framework to develop computer-based techniques. For the internal process there were pandas and NumPy.
Google OCR is mainly used in the text detection part.

Django is used as the backend framework. The special reason of using it is development of all the models are done by python. Using a python-based backend is so easier to integrate. There are so many Python based backends named like Django, Flask etc. But Django is the best option that can be taken as a framework that has the best eco-friendly community. Therefore, that's why selected it.
react.js is used in the backend. Material UI framework has been used for building rich interfaces as its framework.
In the part of the knowledge base and interface engine, there has used the general python framework. There is no other special framework. Pyke is the one which is a tool that builds python math from expert system sells to develop a knowledge base. It is working only for Python version 3.6. But there is 3.8 for these models. If there is 3.6 also there is very poor documentation. General python has been used because of this reason.

7.2.4 Programming Language

Java, Python, and R were among the languages that may be used to build this sort of system. Python was the most popular programming language for creating models and reasoning. Although Python is a general-purpose programming language, it has a rich set of deep learning libraries that may be used for detection parts.

7.2.5 Libraries

OpenCV, PIL, json, NumPy, pandas, time are used in the backend. Material UI used in the frontend.

Libraries	Description	Notable features

OpenCV	It is a free and open-source software library for computer vision and machine learning	Open source / Speed / Integration is simple / Programming easy / Prototyping in record speed.
PIL	A different range of image file formats can be opened, edited, and saved.	The modification of each pixel location. Masking and transparency Management. Sharpening, adjusting the brightness, contrast, or color of an image.
json	Commonly used in web applications for sending data.	Light in weight. Text-based data transmission standard that can be understood by humans. Simple format to deal with.
NumPy	Working with these arrays objects and procedures for processing these arrays are stored in a library.	N-dimensional array object with high performance. Data can be stored in a multidimensional container here.
Pandas	Built in Python for the purpose of data analysis and manipulation	May use it to work with time series data and numerical tables.
Material UI	User interface components can be imported and used in a library.	Because they shouldn't have to start from scratch, the developers save a lot of time this way.

Table 20 - Libraries

7.2.6 IDE

PyCharm IDE is being used to create the backend of the program, while Visual Studio Code IDE is used to construct the whole product, including the front end. Since PyCharm is the most commonly used IDE for Python application development, it was chosen to create the backend of the application. In order to build the entire product, as well as its front end, Visual Studio Code is chosen because it is a web application development tool.

7.2.7 Summary of Technology Selection

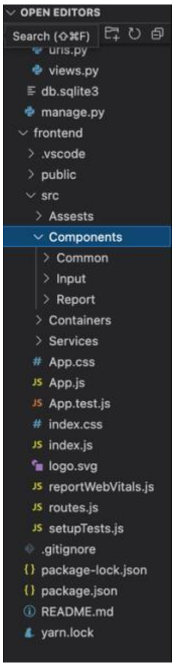
Component	**Tool/ Technology**
Programming language	Python
Development framework	Django, OpenCV
Libraries	OpenCV, PIL, json, NumPy, pandas, Material UI
Frontend framework	Material UI
IDE	PyCharm, Visual Studio Code
Version Control	Git

Table 21 - Summary of Technology Selection

7.3 Implementation of Core Functionalities

There is code based on the material UI from the front-end.

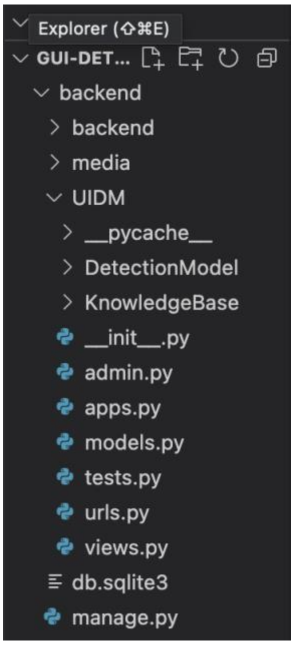
Main file > frontend there are codes related to the front-end.



```
OPEN EDITORS
Search (⇧⌘F)
    urls.py
    views.py
  db.sqlite3
  manage.py
frontend
  .vscode
  public
  src
    Assests
    Components
      Common
      Input
      Report
    Containers
    Services
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
    routes.js
    setupTests.js
  .gitignore
  package-lock.json
  package.json
  README.md
  yarn.lock
```

Figure 16 - Main file > frontend

About the Detection part and .jason file that has generated after detecting the text and components are important as core functions.



```
Explorer (⇧⌘E)
GUI-DET...
  backend
    backend
    media
    UIDM
      __pycache__
      DetectionModel
      KnowledgeBase
      __init__.py
      admin.py
      apps.py
      models.py
      tests.py
      urls.py
      views.py
    db.sqlite3
    manage.py
```

Figure 17 - Detection part and .jason file

Main file > backend > UIDM, in here there is main 2 folders name DetectionModel and KnowledgeBase.

All the detection parts are placed in the detection model. And in the knowledge base there is all the functions that related to the knowledge base.

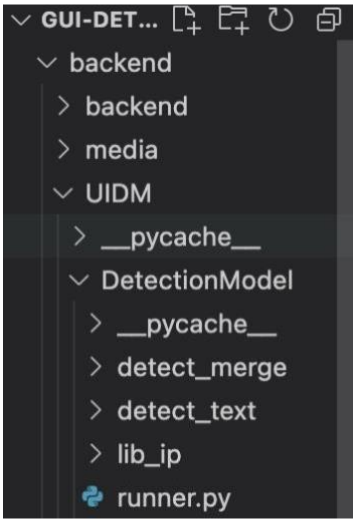


Figure 18 - Main file > backend > UIDM

Here there is text detection part in detect_text.

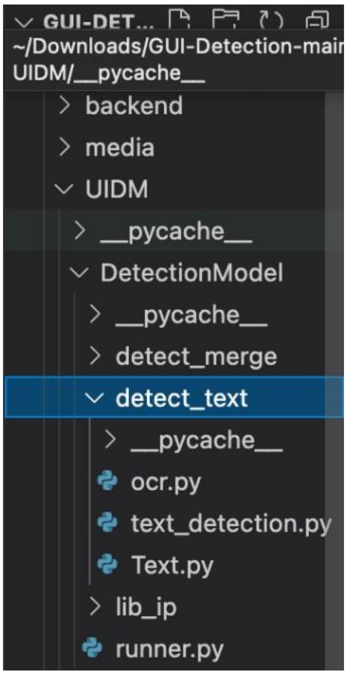


Figure 19 - detect_text

Code that detecting the elements/components is in the detect_merge.

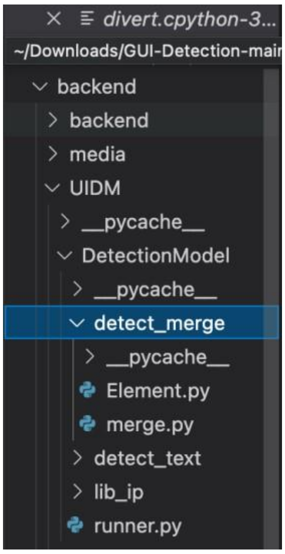


Figure 20 - detect_merge

In the lib_ip, it is merging together the text that has detected from the text detection part and components that has detected from the component detection part. In this case it is not happening just merging and there has created a clean and clear .jason file that has created after removing duplicated component or text if there is.

Here about the knowledge base, the codes are done by using python basics functions. There are separate folders for all the components that has decided to detect the violations.

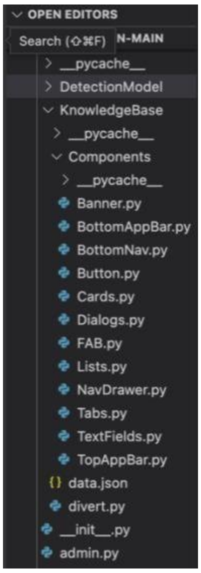


Figure 21 - Clean and Clear .jason file

detect_merge	The place that detects the final image by combining the components that were detected from the component detector and the text component that was detected from the text component.

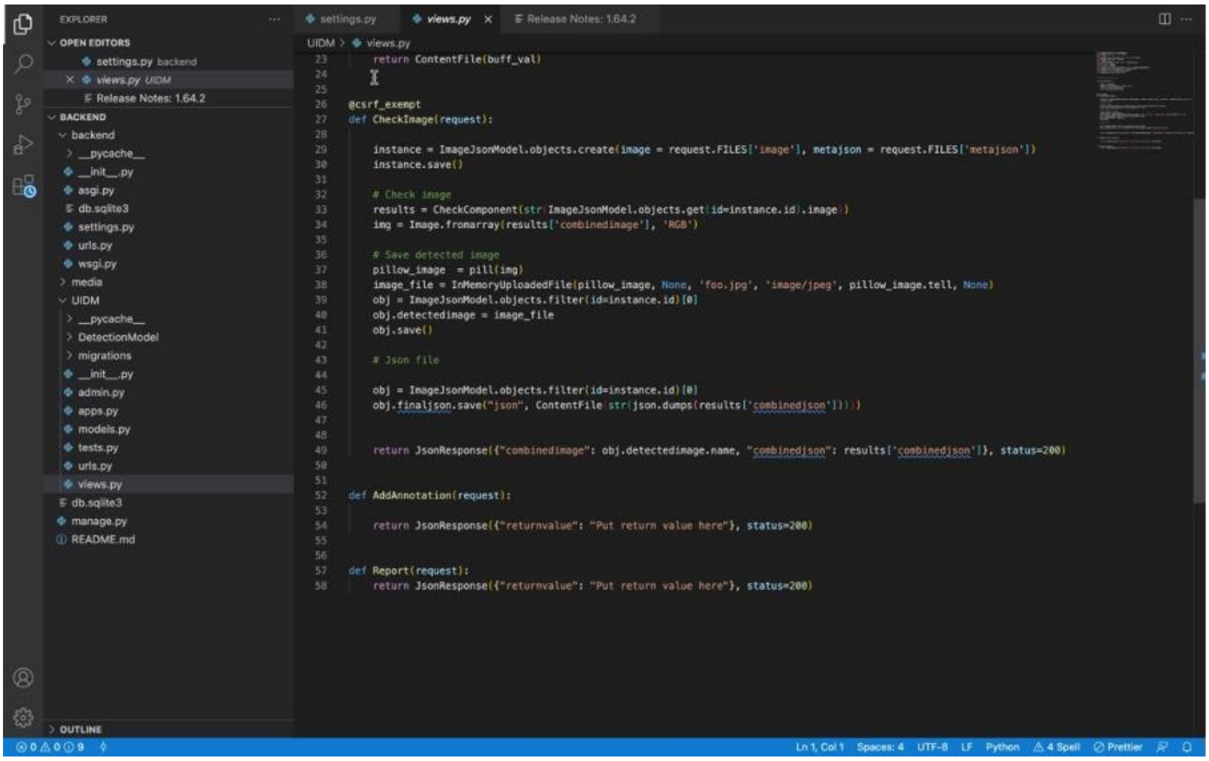
detect_text	The place that text detection files are placed.
lip_ip	The place that exists with the flies that want to detect the components.

Table 22 - Summary of code

Runne.py of the detection model runs all 3 folders and brings the output there.

When taking the knowledge base, there are basically components. Over that, there is a divert.py file. In the component folder, there are separate .py files relevant for all the detected components. Logics that needed to check the rules of each component are in that file. All the components are called from this divert.py file.

View.py file is basically used for code all the views from the backend files.

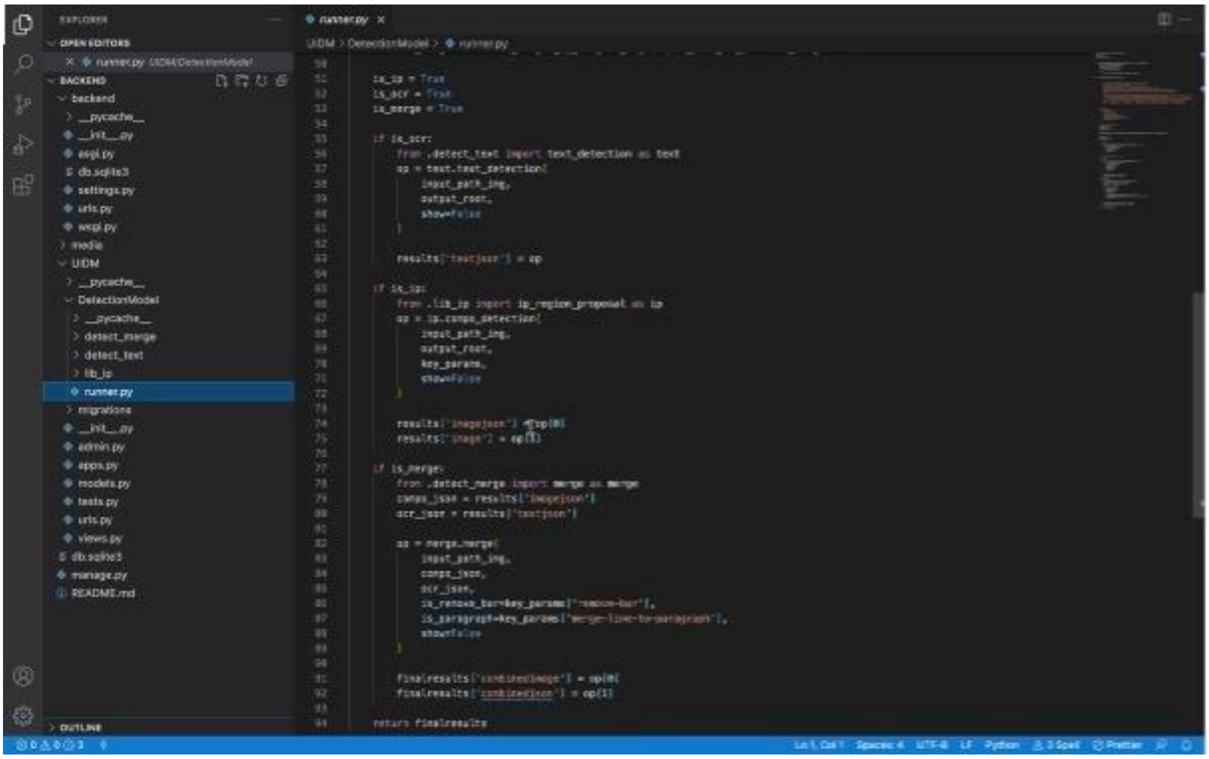


```
        return ContentFile(buff_val)


@csrf_exempt
def CheckImage(request):

    instance = ImageJsonModel.objects.create(image = request.FILES['image'], metajson = request.FILES['metajson'])
    instance.save()

    # Check image
    results = CheckComponent(str(ImageJsonModel.objects.get(id=instance.id).image))
    img = Image.fromarray(results['combinedimage'], 'RGB')

    # Save detected image
    pillow_image  = pill(img)
    image_file = InMemoryUploadedFile(pillow_image, None, 'foo.jpg', 'image/jpeg', pillow_image.tell, None)
    obj = ImageJsonModel.objects.filter(id=instance.id)[0]
    obj.detectedimage = image_file
    obj.save()

    # Json file

    obj = ImageJsonModel.objects.filter(id=instance.id)[0]
    obj.finaljson.save("json", ContentFile(str(json.dumps(results['combinedjson']))))


    return JsonResponse({"combinedimage": obj.detectedimage.name, "combinedjson": results['combinedjson']}, status=200)


def AddAnnotation(request):

    return JsonResponse({"returnvalue": "Put return value here"}, status=200)


def Report(request):
    return JsonResponse({"returnvalue": "Put return value here"}, status=200)
```

Figure 22 - view.py

Here the image file that comes from the request and the meta json file that comes from the request makes an object in the database and saves it.

Checking the components is the first step. The part of detecting the components is done through the Runner.py of this detection part. Detected text is activated through the Runner.py.



```python
    is_ip = True
    is_ocr = True
    is_merge = True

    if is_ocr:
        from .detect_text import text_detection as text
        op = text.text_detection(
            input_path_img,
            output_root,
            show=False
        )

        results['textjson'] = op

    if is_ip:
        from .lib_ip import ip_region_proposal as ip
        op = ip.compo_detection(
            input_path_img,
            output_root,
            key_params,
            show=False
        )

        results['imagejson'] = op[0]
        results['image'] = op[1]

    if is_merge:
        from .detect_merge import merge as merge
        compo_json = results['imagejson']
        ocr_json = results['textjson']

        op = merge.merge(
            input_path_img,
            compo_json,
            ocr_json,
            is_remove_bar=key_params['remove-bar'],
            is_paragraph=key_params['merge-line-to-paragraph'],
            show=False
        )

        finalresults['combinedimage'] = op[0]
        finalresults['combinedjson'] = op[1]

    return finalresults
```

Figure 23 - runner.py

If take the first function,

1. First of all perform the text detection. Then take the result according to it.
2. After that component detection performs and takes the result regarding it.

As the last function,

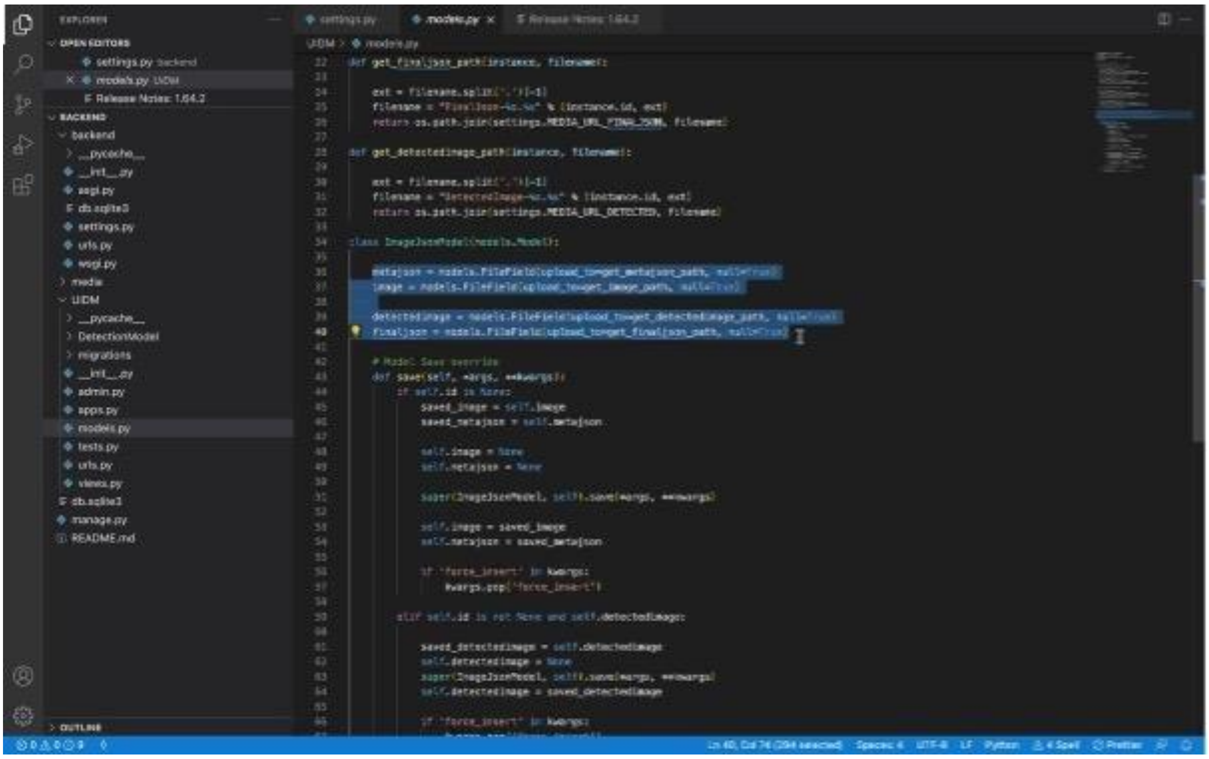
Merge function is performing. It performs on the original image after taking the result of detected component and the result of detected text.
So that there is the combined image and combine json as the output.

Final json and detected image come as the output of this function as basically.

After that again return it to Views.py.

Here the result is taking the things that are returned by the Runner.py.

Returned image converted as an image file (converting through the pythonic pillow library) and saved in the database.
Also received combined json files are saved in the database by converting it to a json file. Then sending that json file and image as a json response to the frontend. So that database is at model.py



```
def get_finaljson_path(instance, filename):

    ext = filename.split('.')[-1]
    filename = "FinalJson-%s.%s" % (instance.id, ext)
    return os.path.join(settings.MEDIA_URL_FINAL_JSON, filename)

def get_detectedimage_path(instance, filename):

    ext = filename.split('.')[-1]
    filename = "DetectedImage-%s.%s" % (instance.id, ext)
    return os.path.join(settings.MEDIA_URL_DETECTED, filename)

class ImageJsonModel(models.Model):

    metajson = models.FileField(upload_to=get_metajson_path, null=True)
    image = models.FileField(upload_to=get_image_path, null=True)

    detectedimage = models.FileField(upload_to=get_detectedimage_path, null=True)
    finaljson = models.FileField(upload_to=get_finaljson_path, null=True)

    # Model Save override
    def save(self, *args, **kwargs):
        if self.id is None:
            saved_image = self.image
            saved_metajson = self.metajson

            self.image = None
            self.metajson = None

            super(ImageJsonModel, self).save(*args, **kwargs)

            self.image = saved_image
            self.metajson = saved_metajson

            if 'force_insert' in kwargs:
                kwargs.pop('force_insert')

        elif self.id is not None and self.detectedimage:

            saved_detectedimage = self.detectedimage
            self.detectedimage = None
            super(ImageJsonModel, self).save(*args, **kwargs)
            self.detectedimage = saved_detectedimage

            if 'force_insert' in kwargs:
```

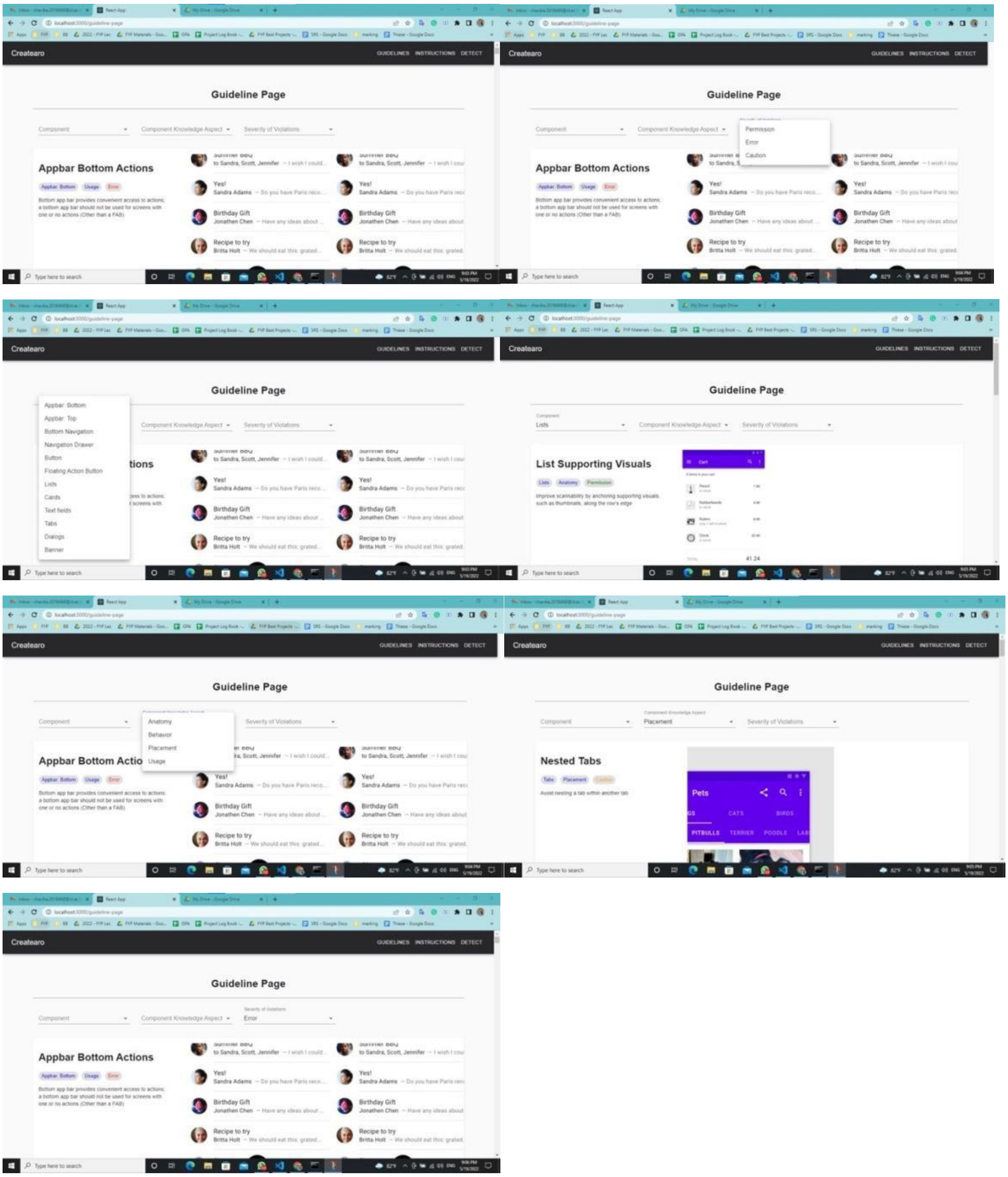
Figure 24 - model.py

There is 4 columns in the database of model.py

1. Meta Json column- once the matajson file came from the request, is saved in the media folder, its path saved by the database.
2. Image column- saving the original image in the media folder and saving the relevant path in the database.
3. Detection image- Once the detected image is saved in the media folder, that file path is saved to the database.
4. Finaljson- save the final json in the media and save its file path in the DB.

7.4 UI - User Interface

7.4.1 Guideline page / Instructions



Appbar Bottom Actions

Bottom app bar provides convenient access to actions; a bottom app bar should not be used for screens with one or no actions (Other than a FAB)

List Supporting Visuals

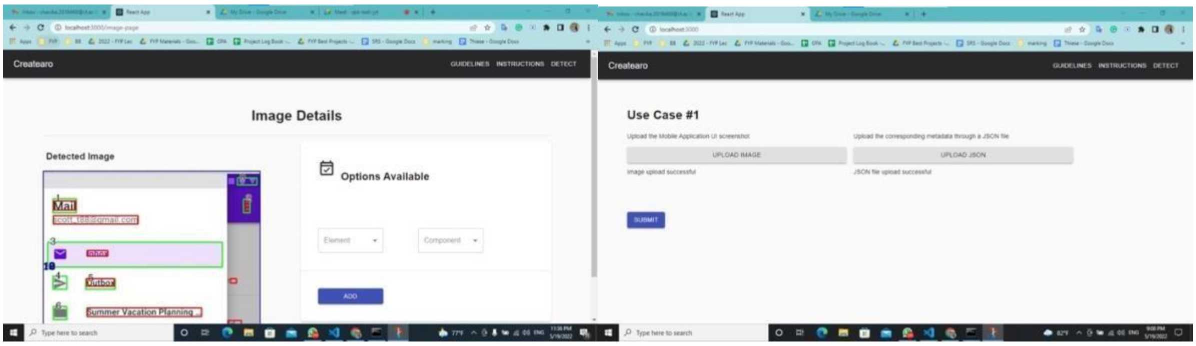
Improve scannability by anchoring supporting visuals, such as thumbnails, along the row's edge

Nested Tabs

Avoid nesting a tab within another tab

Figure 25 - Guideline page / Instructions

7.4.2 Detection



Upload the Mobile Application UI screenshot

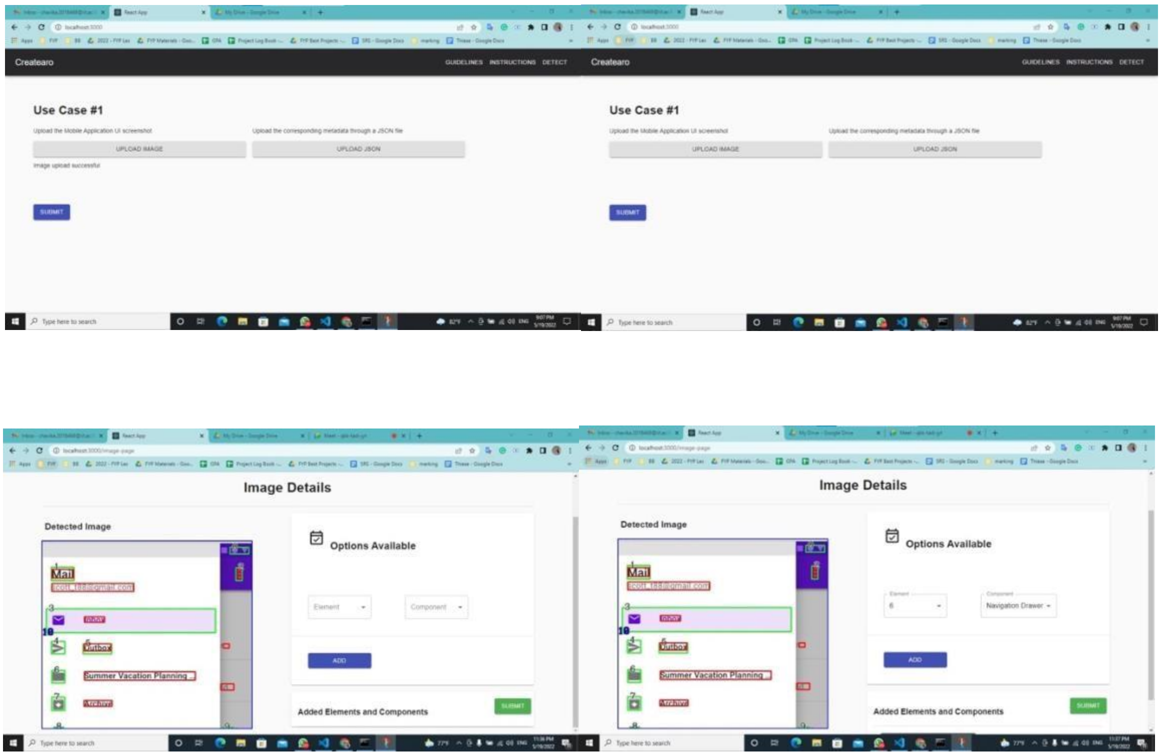
Upload the corresponding metadata through a JSON file

Image upload successful

JSON file upload successful

Figure 26 - Detection

7.4.3 Manual Annotation



Upload the Mobile Application UI screenshot

Upload the corresponding metadata through a JSON file

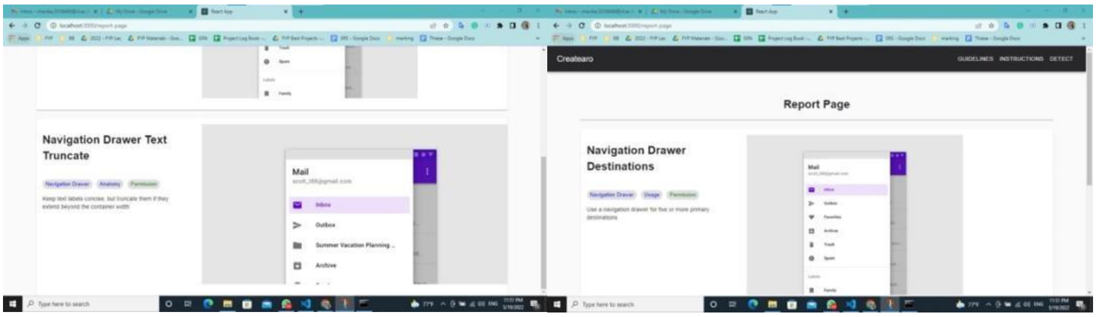
Image upload successful

Element: 6

Component: Navigation Drawer

Figure 27 - Manual Annotation

7.4.3 Report Generation



Navigation Drawer Text Truncate

Keep text labels concise, but truncate them if they extend beyond the container width

Navigation Drawer Destinations

Use a navigation drawer for five or more primary destinations

Figure 28 - Report Generation

7.5 Chapter Summary

This chapter explains why the languages, component tools, and technologies in this chapter were chosen. Code samples and explanations were supplied for the key features.

The essential decisions relating to implementation were supported by arguments. In addition to code explanations, we have included screenshots and screenshots of the Interface tool.

CHAPTER 8: TESTING

8.1 Chapter Overview

Following the discussion of CREATEARO's implementation details in the earlier chapter, this chapter discusses how testing was done to evaluate the predicted flow of functions. This chapter includes testing objectives and procedures such as model validation, benchmark, integration, and testing functional and non-functional testing.

8.2 Objectives and Goals of Testing

The goal of testing is to compare the developed system to the requirements and evaluate whether or not they are satisfied. Among the objectives are :

1. To ensure that the system satisfies the functional requirements.
2. To ensure that the system satisfies the non-functional requirements.
3. To ensure that the code conforms to best practices and effective management.
4. To identify difficulties or defects that were previously uncovered during development. To reach the above goals, the system was subjected to black-box testing, which does not require looking into the core elements that may be used in every level of software testing.

8.3 Testing Criteria

The CREATEARO system was evaluated using two different criterion testing methodologies.

1. Functional Testing -: Evaluating the functional requirements and ensuring that all functions function as expected.
2. Structural Testing -: Evaluating the code to see if it conforms to software engineering industry standards.

8.4 Model Testing

Actually, image processing is the thing that has used here. Libraries are used for it. Therefore, here can’t be do a quantitative model test.

Here the test should be done by changing the parameters. As a example, at the first stage, the elements that don’t need to detect will be detected. Because of the unsuccessfulness of the parameters. By continuing the change of parameters there will be successful result-based vision. And also, that result will be successfully satisfied for the knowledgebase. In that stage author can identify that the success of the violation detected system.

Cant evaluates a quantitative analysis because of its successfulness is finding out by based on the vision.

In the image processing there can be evaluate only a qualitative analysis under the testing phrase. It is describing under the topic ‘8.5 Benchmarking’ with results.

8.5 Benchmarking

Benchmarking should happen according to two sections.

1. **Quantitative Benchmarking**
2. **Qualitative Benchmarking**

But there is no other research or approach before this to detect the iOS violations relevant for this research area which has been selected by the author. So that can’t be able to do a Quantitative benchmark. Relevant for this domain according to the **Qualitative Benchmarking**,

Detection Accuracy - iOS mobile UI mockups and real-world applications
Autor manually inspects iOS UI for violations of HIG guidelines components. Autor finds no violations of the guidelines. This is not surprising given that the 413 UIs were created and designed by iOS mobile professionals to demonstrate best HGI and HCI practices. Even so, Autor uses CREATEARO on this 413 unique iOS UI. Autor have included the original iOS UI to significantly raise the number of test cases and see if createro reports violations that do not exist in such high-quality UI.

CREATEARO scores 0.95. It detects 77 of 80 direct injection design violations and 5 errors. CREATEARO meets 1 of 11 elements and component guidelines. It detects all injected FAB and banner violations. 3 undetected violations due to missing UI components. 5 incorrectly mentioned violations are due to non-existent icons and unusual component distance.

Table 23 - Detection Accuracy - iOS mobile UI mockups and real-world applications

component	**instances**	**Num of instances violations**	**Num of reported violations**	**Precision (%)**	**Recall (%)**	**score**
Button	272	01	04	100	100	1.00
Text Field	182	02	07	100	100	1.00
BottomNav	104	02	06	100	100	1.00
List	80	03	09	78	78	0.78
Nav Drawer	70	01	04	100	100	1.00
Bottom Bar	66	05	16	88	93	0.90
FAB	59	07	21	95	100	0.97
Tab	46	02	06	100	100	1.00
Banner	31	03	09	100	100	1.00
Total	906	26	82	95	0.95	0.95

Table 24 - Testing Result

8.6 Functional Testing

The blackbox testing method was used to check whether the functional requirements (FR) were satisfied as expected.

ID	User Action	Expected Output	Actual Output	Status
FR1	User input valid UI screen	Get attached	Get attached	Passed
FR2	User input valid meta .json file	Get attached	Get attached	Passed
FR3	User input invalid UI screen	Display error	Display error	Passed
FR4	User input invalid meta .json file	Display error	Display error	Passed
FR5	Users select the components and	User let to input elements of a component.	Getting the related components and elements	Passed
	elements that they want to detect.		as an input after giving a numbering system.	
FR6	Submitting the relevant elements and components.	There is an opportunity to user to input components from the detected UI.	Giving opportunity to input the numbers clearly.	Passed
FR7	Studying the violation report.	Display the violations related to inputted UI.	Display the violation rules with samples.	Passed
FR8	Prioritizing the violations.	Principles are cattery as violation levels.	Display the principle as warning, permission and error.	Passed

FR9	Visit interactions page, to identify the workflow	Giving a help to the user to handle the system.	Give guideline to the flow.	Passed
FR10	Principles of all the aspects applicable for a specific component.	Display the filtered HIG principles significant for the components.	Filtering successfully.	Passed
FR11	Principles of all the components relating to a specific aspect.	Filter and display HIG principles according to the relevant aspect.	Filtering successfully.	Passed
FR12	Severity of violations	Display the HIG principles after filtering according to the severity.	Filtering successfully.	Passed

Table 25 - Functional Testing

8.7 Module and Integration Testing

Module	Input	Expected Output	Actual Output	Status
Input images for detecting violations	Correct format	Attached	Attached	Passed
	Wrong format	Display error	Display error	
Component detection part	Input application UI/Component	Detected	Detected	Passed

Test detection part	Input application UI/Component	Detected	Detected	Passed
Elemaned detected output	iOS application UI	UI that has detected all the elements clearly.	UI that has detected elements. But it does not detect only the main elements of the UI. All the elements are detecting	Passed
Detected violation report	Tested violated iOS UI	Display the detected violations with its HIG principles.	Output has come as expected. Priorities those principles as well.	Passed

Table 26 - Module and Integration Testing

8.8 Non-Functional Testing

Non-functional evaluation was performed in conformance with the non-functional requirements set out in chapter 4.

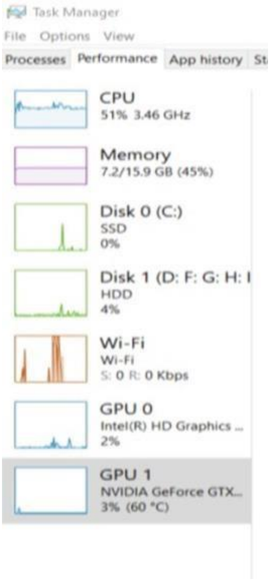
8.8.1 Performance

8.8.1.1 Model Testing

Most important was how the established research component performed in the final product. Therefore, it was important to check the product in a resource-constrained local environment. Tested with i5-1035GI CPU and 16GB RAM.

8.8.1.2 GPU and Memory Performance while Testing

Because segmentation models mostly run-on edge devices, measuring GPU, memory, and CPU performance is critical. 51% CPU, 45% memory, 3% GPU.



CPU
51% 3.46 GHz

Memory
7.2/15.9 GB (45%)

Disk 0 (C:)
SSD
0%

Disk 1 (D: F: G: H: I
HDD
4%

Wi-Fi
Wi-Fi
S: 0 R: 0 Kbps

GPU 0
Intel(R) HD Graphics ...
2%

GPU 1
NVIDIA GeForce GTX...
3% (60 °C)

Figure 29 - GPU and Memory Performance while Testing

8.8.1.3 GUI Tool Performance

Lighthouse in Google Chrome tested GUI performance, quality, best practices, responsiveness, accessibility, and SEO. This image shows train-and-evaluate user interface reports.

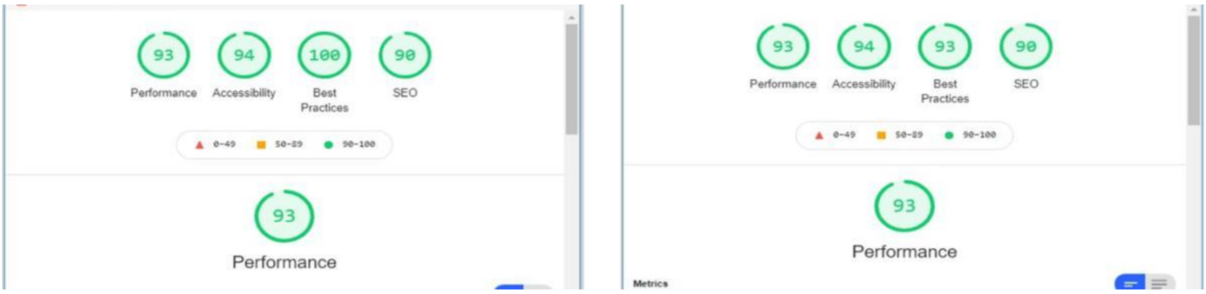


Figure 30 - GUI Tool Performance

8.8.2 Usability Testing

The prototype's usability was tested in a focused group setting. As the focused group, the same evaluation target audience was used.

Question – no.1	How do you like the prototype's usability? 1- Poor 5- Excellent
Aim	To get feedback about the usability of CREATEARO’s Prototype.

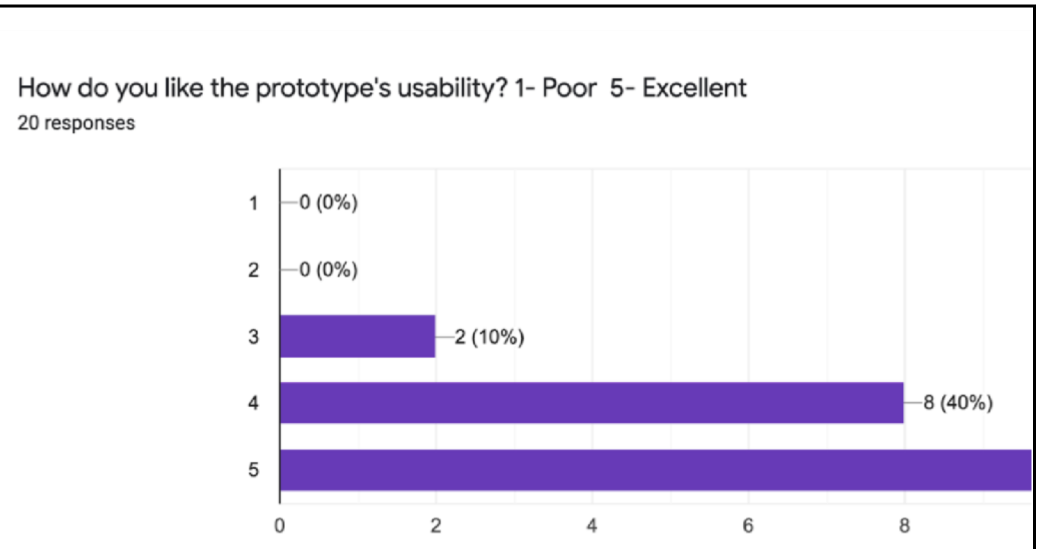
Findings	 Got 20 Responses from the audience. 10 from it, that means 50% has evaluated that it has excellent usability. And from the other 50%, 10% Is has evaluate it as neutral and 40% is evaluating it as good usability.

Table 27 - Usability Testing

8.8.3 Maintainability Testing

Focused group testing of maintainability the focus group was the same evaluation target audience.

Question – no.1	What will be the code quality score? 1- Poor 5- Excellent
Aim	To evaluate that there is a satisfied or not code quality.

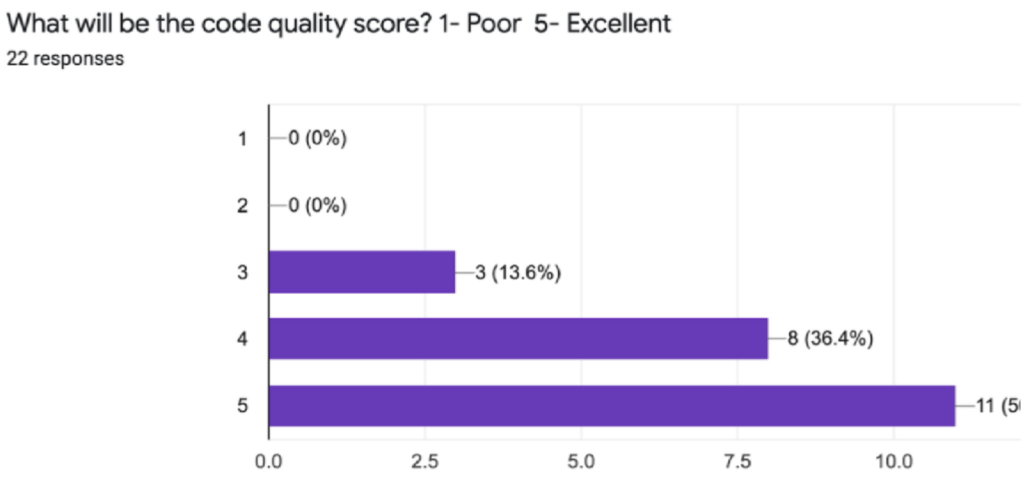
Findings	 What will be the code quality score? 1- Poor 5- Excellent 22 responses
	According to the Feedbacks 50% has rated the code quality as good.

Table 28 - Maintainability Testing

8.9 Limitation of Testing Process

This output can be obtained as the limitations of the testing process. There, can't get a quantity from testing because it is image processing. Here the testing is happening mainly based on the vision. And if there is a satisfied result according to the changes of parameters, that means if that phrase detects the violations correctly it can be known as a successful phrase. When studying the evaluation part of chapter 9 it can suggest that there is a stage to satisfy the current stage.

But, continuous changes of these parameters, there is an ability to show a 100% success result more than this from each component.

8.10 Chapter Summary

This chapter presented the testing's goals and objectives. To achieve the evaluation goals and objectives, functional, non-functional, integrated, and model evaluations were conducted. The system was then benchmarked using the best-performing models generated, including a general solution that generalizes well across various options.

In addition, functional, implementation, and non-functional tests have been performed to ensure that the application and research meet the defined guidelines and perform as expected.

CHAPTER 9: EVALUATION

9.1 Chapter Overview

This chapter provides a review of the feedback received from experts (domain, technical, researchers, and supervisors) on many aspects of the research and prototype, such as concept novelty, and end-product quality, and so on. The author's evaluation is also offered.

9.2 Evaluation Methodology and Approach

A project's success can be measured through evaluation. The research domain, problem, design, testing, and other key aspects of a project were all submitted for feedback.

The research's evaluation process included both a quantitative and qualitative approach. According to the previous chapter, the research result was evaluated to other reliable and novel domain research using quantitative metrics.

Using thematic analysis, the aim here is to conduct a qualitative evaluation based on feedback given by experts.

9.3 Evaluation Criteria

Criteria	Target
Problem domain	Current relevance. Ensure that there is suitable depth for a research project that will contribute to solving the research gap in the domain.
Research Gap	To evaluate how demanding the project
Tested architectures and current architecture	Identify the most relevant architecture based on the result. It was evaluated from time to time.
Literature of theory	To verify that a thorough literature review of the domain and relevant technologies has been completed before finding a gap in the domain.
GUI	Overall feedback about user experience.

Conclusions and Future works	Suggestions/Improvements for architecture according to the upcoming technologies and methods.
Prototyping	Considering about research domain, prototyping and user experience of the system must.
Evaluate the approach	Evaluate how the research gap has solved the research gap with synthetic data while real-world UI screens are limited and evaluate that technique under current conditions.
Benchmarking	To verify the metrics used to evaluate the model, as well as how it performed in contrast to other methods in the domain.

Table 29 - Evaluation Criteria

9.4 Self-Evaluation

The author decided to do an analysis of the research's themes. Self-evaluation was also decided to be carried out using the previously defined evaluation metrics.

Criteria	Self-evaluation
Problem domain, the research gap, and its depth	According to the author, this approach is really successful in detecting UI component features. And if this system can identify UI component, it has completely filled the gap. But here author using manual annotation part for that. It also better for that.
Methodology, Literature of theory	To begin, a thorough literature review was required to identify any gaps in the domain. Once the research gap had been identified, the literature had to be searched for techniques that may be used to fill it. As a result, component violation detection techniques were explored, and a possible technique was identified. A thorough approach was followed in each and every part of the project over its entire life cycle.

Architecture and Implementation	Due to the difficulty of the concepts and components research-based, the architecture, and execution were complicated. The functional and non-functional requirements, as well as the main components, were identified through regular discussions and prototypes, as well as interview methodologies. The development was carried out using cutting-edge technology. Throughout the development and implementation phase, code quality and industry standards were taken into account. The features, design/architecture, and implementation were all completed to a decent quality.
GUI and Prototyping	The prototype was created as a web application. The components were well-designed and might be improved further. The product enhancements were included in the section for future work. The prototype's usability.

Table 30 - Self-Evaluation

9.5 Selection of the Evaluators

The research interests and expertise areas of the domain and technical experts were evaluated. When identifying the evaluators, knowledge in pertinent fields was evaluated. Experts from UI/UX domain, including front-end developers, were selected for the Technical Expert category.

	ID	**Position**	**Academic**
Domain Expert - 01	EV01	Senior Research Scientist	Ph.D.
	EV02	iOS Developer and ASO	Ph.D.
	EV03	Senior Software Engineer	BSc
	EV04	Senior iOS Developer	BSc
	EV05	Associate iOS Developer	MSc

	EV06	Senior Software Engineer	MSc
	EV07	Senior Mobile Developer	MSc
	EV08	iOS Developer	BSc
	EV09	ASO	BSc
Technical Expert - 02	EV10	Associate UI/UX Lead	BSc
	EV11	Ph.D. Candidate	Ph.D.
	EV13	UX Consultant	MSc
	EV14	UX Consultant	Ph.D.
	EV15	UI/UX Engineering	BSc
Beginner Expert - 03	EV16	Undergraduate	UG
	EV17	Internship student (Undergraduate)	UG

Table 31 - Selection of the Evaluators

9.6 Evaluation Results

In the research evaluation phase uses qualitative and quantitative methodologies.

9.6.1 Qualitative Result Analysis

Deductive thematic analysis was used to conduct qualitative result analysis.

	Evaluation Summary and Feedback
Problem domain, the research gap, and its depth	

Domain Expert - 01	The identified gap in the literature is relevant to the component detection and identification part, and indicates a research domain limitation. This system is very useful and important for ASO, iOS developers, and UI designers. For undergraduate research, the study has enough depth.
Technical Expert - 02	The complexity and the depth of the research should be appreciated, according to evaluators. It's important to do research for future projects, and it's been narrowed down to a specific domain.
Beginner Expert - 03	The author has chosen a domain for her project that is both relevant and of high interest to academics and industry.
Methodology, Literature of theory	
Domain Expert - 01	Before going to come up with an approved methodology, the author seems to have completed a detailed literature review and thoroughly explored the problem domain. A literature review was done on a large variety of models, and the expertise was high.
Technical Expert - 02	The methodology has handled the research challenge and done a great job. There are always more things to explore, such as component classification or other architectures, but the results of this methodology seemed to be acceptable.
Architecture and Implementation	
Domain Expert - 01	The suggested solution architecture and technology selection are both suitable. Both in terms of research and implementation, this is a challenging domain. However, the author did this project justice.

Technical Expert - 02	The recommended layered architecture is in line with the project concept. The project's implementation is competent and exceeds industry standards; it uses cutting-edge technology, and the software maintenance looks to be acceptable.
Beginner Expert - 03	Extremely useful for iOS developers. Also from their viewpoint, this helps all stages in the software life cycle.
GUI and Prototyping	
Domain Expert - 01	The given GUI and Prototyping were appreciated by the evaluators. A good option was found to be incorporating qualitative analysis into the GUI.
Technical Expert - 02	This research topic, project structure, achievements and output, and GUI are all noteworthy.
Beginner Expert - 03	Best user experience (UX).

Table 32 - Qualitative Result Analysis

9.6.2 Quantitative Analysis Analysis

The quantitative analysis was used to evaluate quantitative data. The quantitative evaluation was divided into 4 components.

1. Problem domain, the research gap, and its depth.
2. Methodology, Literature of theory.
3. Architecture and Implementation.
4. GUI and Prototyping.

9.6.2.1 Problem domain, the research gap, and its depth

Question – no.1	Research novelty. 0 – No novelty, 5 – Very high

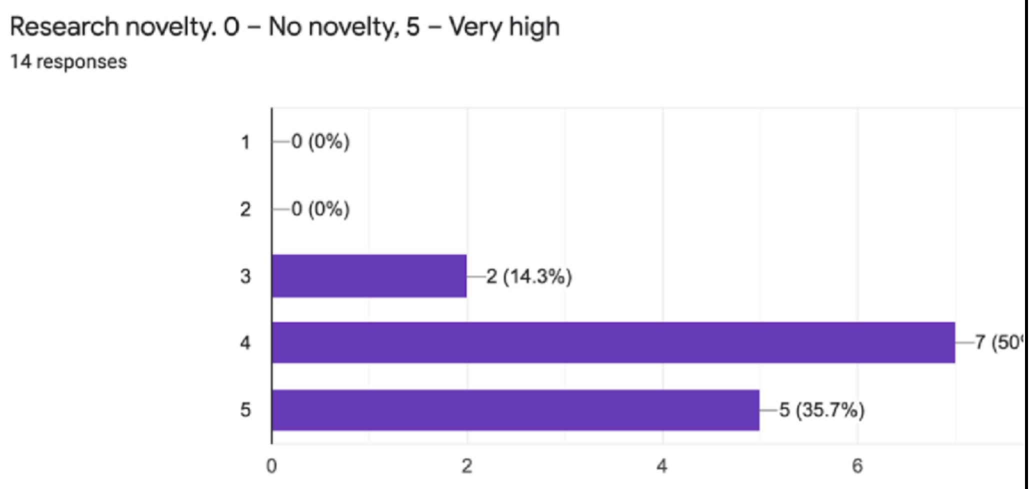
Aim	Get feedback about project novelty to get as a quantity of its success.
Findings	 35.7% of the evaluators gave 5 stars. For novelty, the majority of 7 evaluators (50%) gave this study a 4 out of 5 rating. 2 of the evaluators (14.3%) said they were satisfied. The novelty of the research area was praised by all of the evaluators. Hence, this is a novelty.

Table 33 - Problem domain, the research gap, and its depth - Question-no.1

Question – no.2	The quality of the scope of the project as undergraduate students. 0 – Not satisfied, 5 – Very high
Aim	Got feedback from evaluators to know if the project scope is enough.

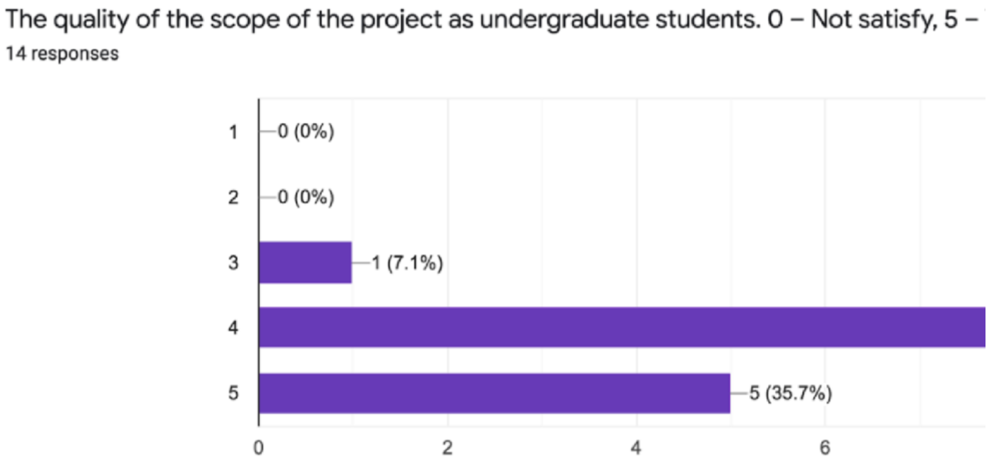
Findings	 This research has a good depth for the undergraduate level, according to 35.7% of the evaluators. It has a high depth, according to 57.1% of respondents. The depth rating of 7.1% is satisfactory. The evaluators were all satisfied with the breadth of both the project scope.

Table 34 - Problem domain, the research gap, and its depth - Question-no.2

Question – no.3	How difficult is it at the undergraduate level? 0 – Very Low, 5 – Very High
Aim	After identifying if there is a change of research paper according to undergraduate level, find out that there can be added a value to the research.

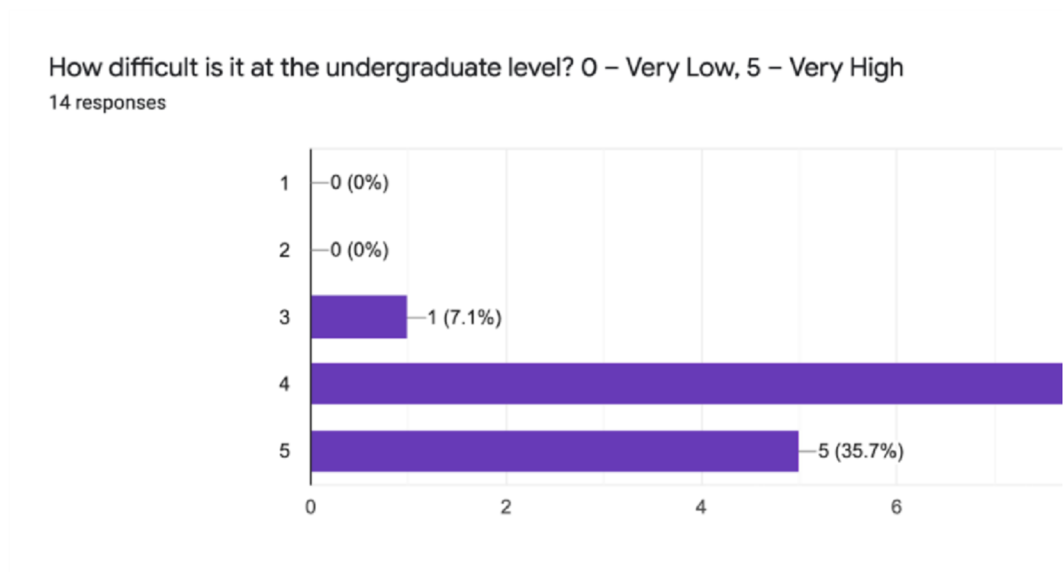
Findings	 It received a very high challenging scope rating from 35.7% of evaluators. 57.1% thought it has a good difficulty. 7.1% of those surveyed rated it as medium or satisfactory. most of the evaluators did agree that the research project has a good difficulty level for an undergraduate.

Table 35 - Problem domain, the research gap, and its depth - Question-no.3

Question – no.4	The project's technical complexity. 0 – Very Low, 5 – Very High
Aim	To get feedback about the proposed techniques.

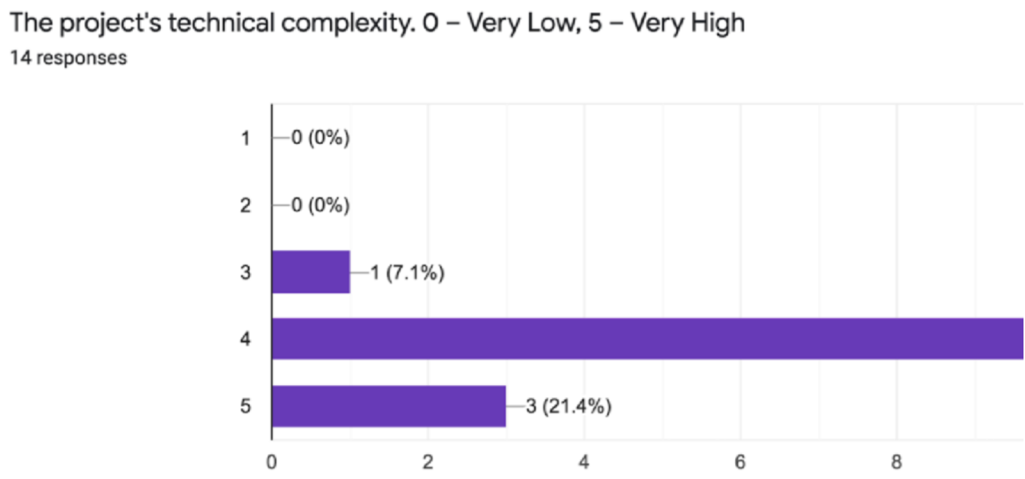
Findings	 The technical difficulty was rated as very high by 21.4% of the evaluators. The technical difficulty was rated as high by 71.4% of the evaluators. 7.1% thought it was medium. All of the evaluators did agree that the technical difficulty of the proposed approach is acceptable.

Table 36 - Problem domain, the research gap, and its depth - Question-no.4

9.6.2.2 Methodology, Literature of theory

Question – no.1	What is the architecture of the system? 1 - Poor 5 - Excellent
Aim	To get an idea about the architecture.

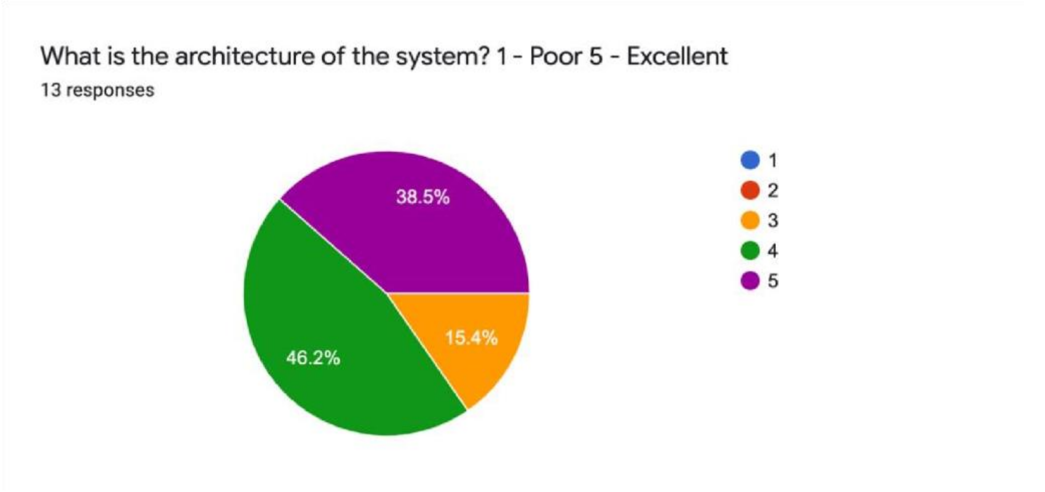
Findings	 38.5% of the evaluators gave the highest rating of 5 out of 5 stars. The most of of the evaluators (46.2%) gave this study a 4 out of 5 rating. One evaluator (15.4%) received a 3 out of 5 rating. All of the evaluators have been satisfied with the system architecture provided.

Table 37 - Methodology, Literature of theory - Question-no.1

Question – no.2	The stack of technologies. 1 - Outdated 5 - Latest
Aim	To find out that the used techniques are matching with the current up to date technologies.

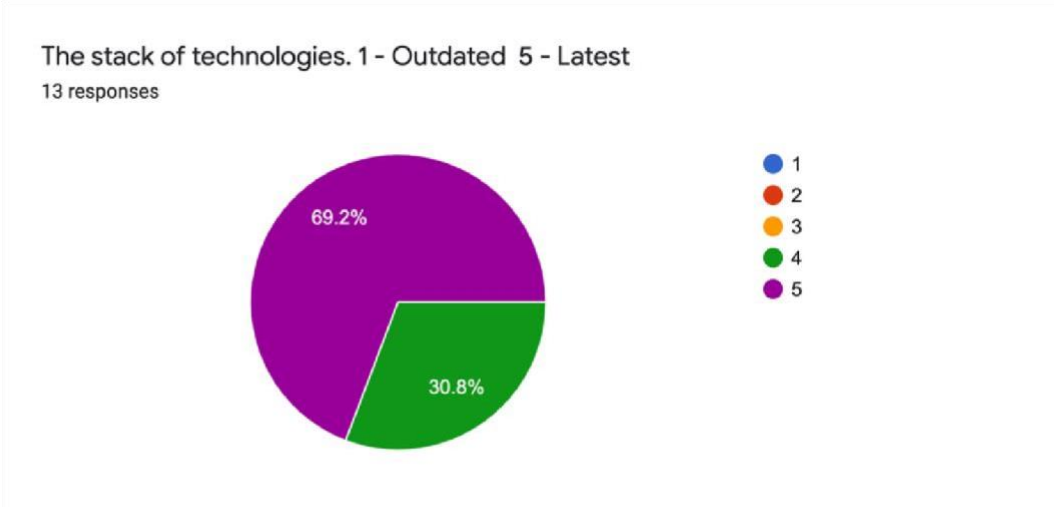
Findings	 The tech stack earned a 5 out of 5 rating from the majority of the evaluators (69.2%). The evaluators agreed that the technology stack in use is up to date.

Table 38 - Methodology, Literature of theory - Question-no.2

9.6.2.3 Architecture and Implementation

Question – no.1	Is the presented solution comprehensive in its approach to the problem? 1 - Strongly Disagree 5 - Strongly Agree
Aim	To get feedback about solution comprehensive in its approach to the problem and to identify the things that wanted to improve.

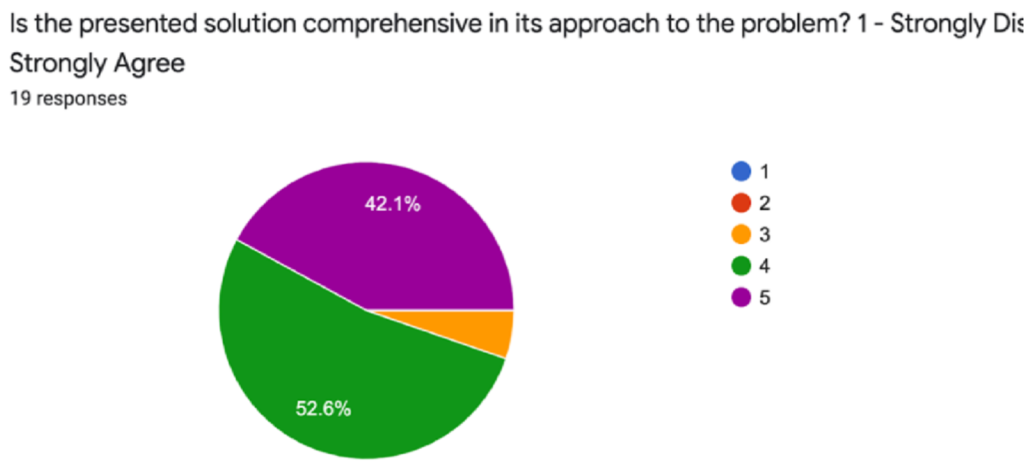
Findings	 Is the presented solution comprehensive in its approach to the problem? 1 - Strongly Dis Strongly Agree 19 responses The statement was strongly agreed upon by 42.1% of the evaluators. The majority of the evaluators 52.6% rated as they agree with the statement. 5.3% were neutral. With the exception of two evaluators who rated CREATEARO as neural, all evaluators agreed that CREATEARO has a depth of solving the identified problem.

Table 39 - Architecture and Implementation - Question-no.1

Question – no.2	Do you believe the system offers an alternative to the existing problem? 1 - Strongly Agree, 5 - Strongly Disagree
Aim	To find out that detecting this violation is useful for the designers and developers.
Findings	

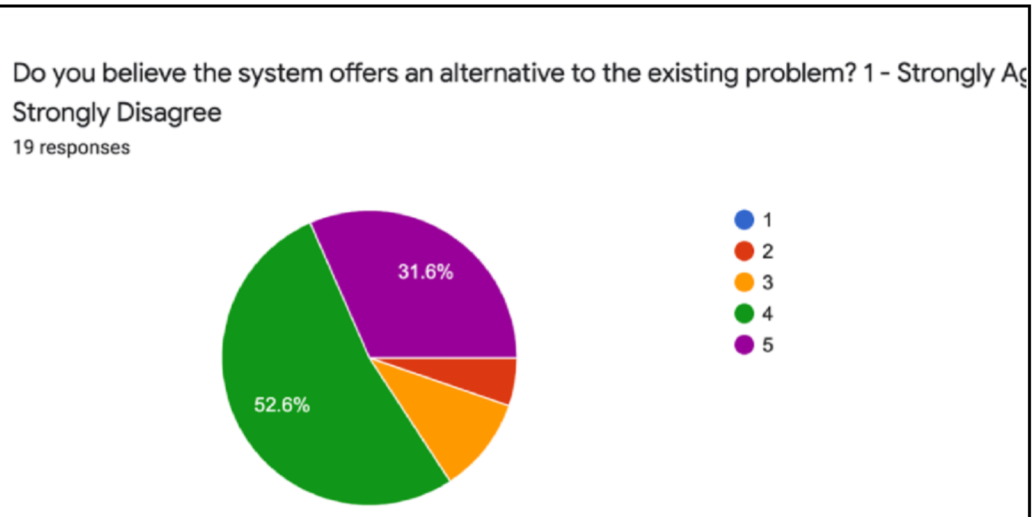
	 Do you believe the system offers an alternative to the existing problem? 1 - Strongly Ag Strongly Disagree 19 responses Nearly half (52.6% = 10) of the participants rated themselves as strongly agreeing with the statement. 31.6% of the people who took the survey agreed with the statement in its entirety. CREATEARO was widely regarded as a viable solution to the identified problem by the evaluators.

Table 40 - Architecture and Implementation - Question-no.2

9.6.2.4 GUI and Prototyping

Question – no.1	The solution's presented graphical user interface? 1 – Poor, 5 – Outstanding
Aim	This evaluation is useful for the project. Because of detecting violations of the research of the author. It should be neated if it is also a desktop app. Therefore, get this feedback to get an idea of it.

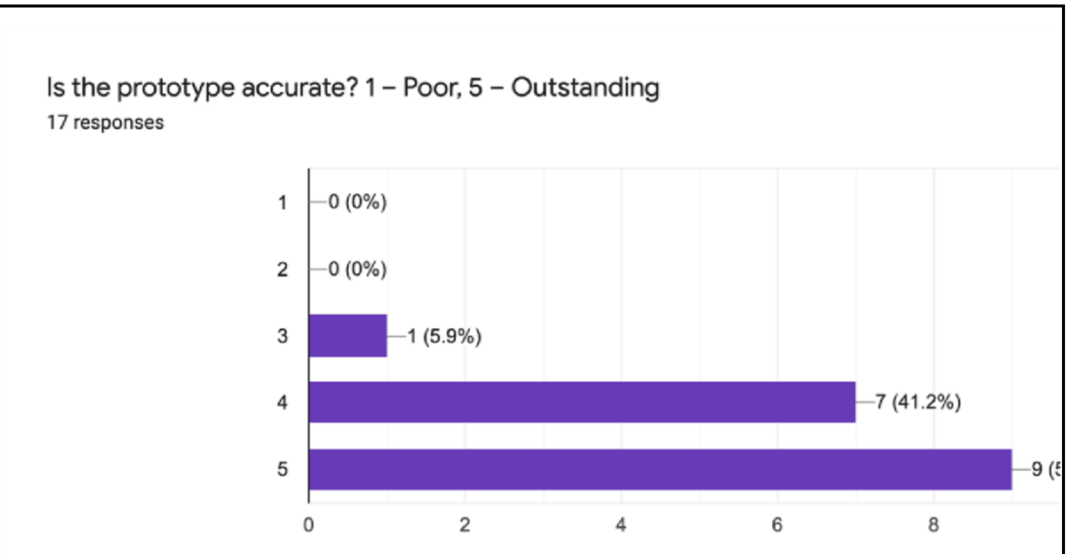
Findings	 52.9% of evaluators rated the GUI as excellent. 41.2% of people thought it was a good GUI solution. 5.9% of them were classified as neural. The GUI was evaluated by UI/UX experts as well as domain/technical experts. The CREATEARO will be used by domain/technical experts. Almost all of the evaluators gave the GUI-based solution positive feedback.

Table 41 - GUI and Prototyping

9.6.2.5 Testing, Accuracy of the Prototype

Question – no.1	Are the evaluation metrics used suitable? 1 - Strongly Agree, 5 - Strongly Disagree
Aim	There is no evaluation to get feedback. See if it is relevant with the research.

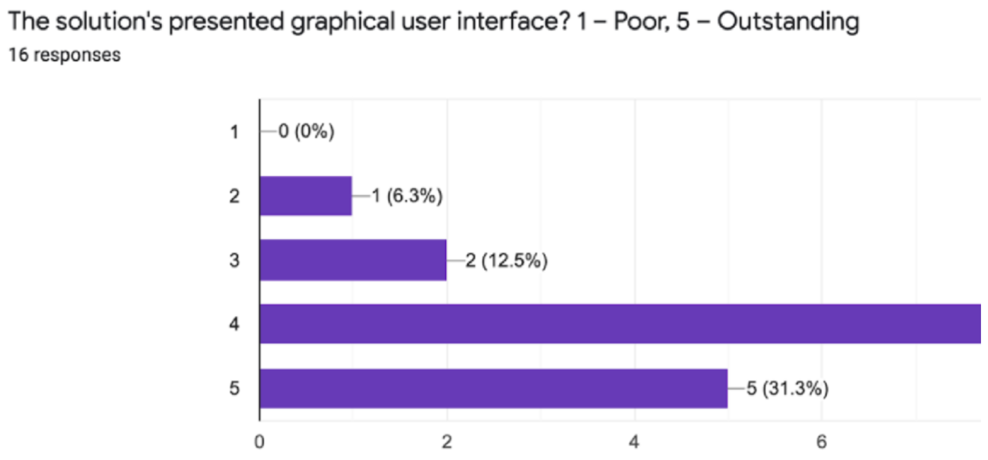
Findings	 31.3%of the evaluators rated the statement as strongly agreeing. 50% of the evaluators agreed with the statement. According to 1 evaluator, the responses disagreed, and 1 evaluator, the responses were neutral. These are the most common evaluation metrics in the domain. The majority of evaluators (81.3% = 13) agreed that the measurement methods used were essential to the research.

Table 42 - Testing, Accuracy of the Prototype - Question-no.1

Question – no.2	Is the prototype accurate? 1 – Poor, 5 – Outstanding
Aim	To get an evaluator idea about the prototype.

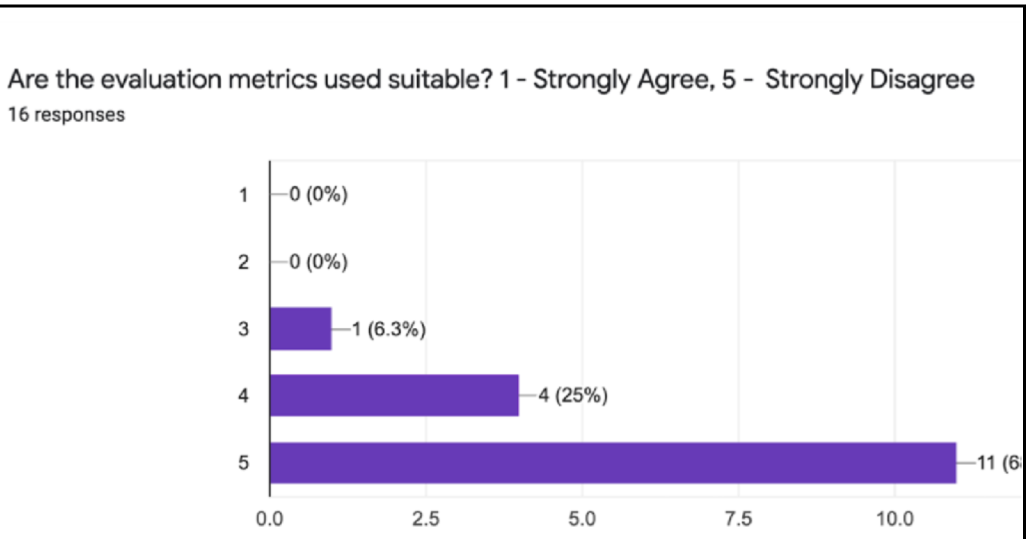
Findings	 The prototype's accuracy was rated excellent by 68.8% of the evaluators. The majority of the evaluators (25%) rated CREATEARO as having good accuracy. One evaluator rated it as average, while another rated it as poor. As a result, 73.8% of the validators were pleased with the CREATEARO's accuracy.

Table 43 - Testing, Accuracy of the Prototype - Question-no.2

9.7 Limitations of Evaluation

As requested by the evaluators, a complete report covering design, implementation, testing, GUI functionalities and prototyping was provided, along with a Google form to collect feedback.

9.8 Evaluation on Functional Requirements

The MoSCoW approach was used to rank the relevance of system needs and assign them a priority level. (IM - Implemented, NIM - Not Implemented)

ID	Requirement and Description	Priority Level	Evaluation
FR1	Users must be allowed to use or input in images to be detected and additionally user can input meta data also.	M	Implemented
FR2	The front-end must validate the image file type input.	M	Implemented

FR3	The model must identify components and text on the UI image using detection model.	M	Implemented
FR4	Give users a chance to do a manual annotation with clear interface.	M	Implemented
FR5	Need to give a numbering to the detected UI elements and identify what the user wants to input.	M	Implemented
FR6	And leave a space in the UI for the user to input the component number they want to identify violations.	M	Implemented
FR7	Need a UI that can see identifiable violations.	M	Implemented
FR8	Prioritize detected Violations.	M	Implemented
FR9	Without an input image, the user should be able to see the rules that apply to the design aspects and design dimensions of a component in general.	M	Implemented
Non-functional requirement completion rate = (9/9)*100%			100%

Table 44 - Evaluation on Functional Requirement

9.9 Evaluation on Non-Functional Requirements

ID	**Requirement and Description**	**Priority Level**	**Evaluation**
NFR1	The system should be able to operate without lagging or crashing when images are input and processed.	M	Implemented
NFR2	Need to have a GUI to have the best user experience that is easily recognizable.	M	Implemented

NFR3	Even a fresh user who needs a user instruction guide should be able to easily handle the tool.	M	Implemented
NFR4	The output image given to the user for manual annotation should be clear.	M	Implemented
NFR5	Follow coding best practices and standards.	S	Implemented
NFR6	The system should generate relevant error messages.	S	Implemented
Non-functional requirement completion rate = (6/6)*100%			100 %

Table 45 - Evaluation on Non-Functional Requirements

9.10 Chapter Summary

The methodology and approach used in the evaluation were fully detailed in this chapter. Different evaluation criteria were established and dividing the research domain into different parts. The self-evaluation also based on this.

A thematic analysis was conducted based on expert opinion collected through interviews and questionnaires. The result is summarized in the chart according to the thematic analysis.

CHAPTER 10: CONCLUSION

10.1 Chapter Overview

This chapter discusses the overall research's concluding observations. The project's initial targets, priorities, research objectives and learning outcomes are discussed, as well as the challenges that were encountered. Utilization of skills and knowledge gained over four years, as well as newly learned skills through research, deviations from the initial proposal, limitations, suggestions for further research, and ability to contribute to the knowledge base are all discussed here.

10.2 Achievement of Research Aims & Objectives

10.2.1 Aim of Research

The goal of this research project is to design, develop, test and eveluate a framework that will allow detection and identify iOS UI violations.

10.2.2 Research Objectives

The following table describes the progress of the research objectives toward project completion. The following LOs were met in regard to each research objective.

Research Objective	Status	Learning outcomes
Literature Review - A review of research methods and systems.	Completed	LO1/LO2/LO4/LO5/LO8
Requirement Analysis - Using multiple techniques to gather requirements.	Completed	LO1/LO2/LO8
System Design and Architecture - Perform the necessary system design and generate the system design specification.	Completed	LO1/LO2/LO5/LO8
Implementation - Using the factors identified during the requirement analysis of structures phases, develop the system.	Completed	LO5/LO8
Testing and Evaluation - Testing and evaluating the mockup to ensure that the system functions properly.	Completed	LO4/LO5/LO7/LO8

Table 46 - Research Objectives

10.3 Utilization of Knowledge from the Course

Module	Description
Python OOP	These modules provided important programming knowledge for OOP, beginning with the fundamentals. Although OOP was just not used during the software development cycle, it was valuable in developing certain prototype decisions.

Web Design and Development, Server- side Web Development	These modules provided an introduction to client-server architecture and advanced web frameworks, beginning with HTML, CSS, and JavaScript fundamentals. The knowledge gained from these modules was crucial for implementing the application's front end.
SDGP	This is the module that started this research. It required identifying, designing, developing, and testing a prototype.
Algorithm: Theory Design and Development	This module introduced the basics of ML and Data science. This research's development phase is based on algorithms, search strategies, and graphs. Using CREATEARO was beneficial.

Table 47 - Utilization of Knowledge from the Course

10.4 Use of Existing Skills

A variety of skills were useful for the development of the system throughout the degree program. However apart from that, the mentioned procured skills greatly helped me in completing the project.

Machine learning and deep learning – The Coursera and LinkedIn courses that followed greatly helped in filling the knowledge gap.

Python programming – The author's previous background experience of the programming language enabled author to begin development concurrently with the other chapters.

Use python for knowledge base functions.

10.5 Use of New Skills

During the course of the project, author learned the following skills.

1. **Image processing part**
2. **Add filtering options for rules in knowledge base**
3. **File upload option through interface** - Displaying the component detected UI for the user to do the manual annotation of the interface again.

4. **Writing skills for research papers** – The author was able to publish three papers while working on the final year research project. With the project completion, I gained skills for contributing to the research community.

Additionally, time and project management, and writing skills were obtained.

10.6 Achievement of Learning Outcomes

What was Learnt	**LO/s**
This research gave the author the ability to accurately research domain in a topic of focus, as well as critically evaluate works in the area.	LO1/ LO2/ LO4
To be successful, every step of the project had to be planned ahead of time, mitigating issues. Time management and proper planning were critical.	LO3/LO7
The author was completely new to the research domain of text and component detection base computer vision base which meant that author had to learn most of the technical skills from the ground up. The degree of difficulty was massive.	LO5/ LO6
Throughout the module, each action had to be thoroughly documented(Proposal/PSPD/FYP), and documentation skills were extremely important in writing theses.	L08
The module improved requirement gathering skills. Formal questionnaires were created, formal interview scripts were written, and planned interviews were properly conducted. These abilities were very useful in putting together such a useful initial design and will be important in the future too.	LO2/LO4
Testing, system failing, and attempting again defined the entire development and testing phase of the research.	LO5/LO6

Experts were required to evaluate the prototype after its implementation, giving their viewpoints and evaluation feedback based on the author's presentation of the project's concept, implemented solution, and achieved results.	L09

Table 48 - Achievement of Learning Outcomes

10.7 Problems and Challenges Faced

Problem/Challenge	Solution
There are multiple disciplines involved in research.	The identified domain existed in the field of Computer Vision, involving the application of UI detection techniques in combination with image generative models, such as text detection and component detection. As a means of overcoming this challenge, the author had to conduct extensive research on each domain and dedicate an important number of hours to perfecting deep learning skill parallelly with the start of the project timeline.
Extensive project scope.	As the de-raining domain was expansive, with subdomains including UI violation and detection, it was a challenge for the author to sufficiently scope the project to a manageable margin.
Innovative generative models entail a steep learning curve.	It was difficult to understand how various generative models operate and to apply them. As soon as the work plan has been finalized, tests with the technology demonstrator were initiated in order to overcome this obstacle.
Superior hardware requirements.	Due to the resource-intensive character of the models, highend hardware was required to run them efficiently. To address this difficulty, a laptop with powerful CPU and GPU abilities was gained.

Table 49 - Problems and Challenges Faced

10.8 Limitations of the Research

1. Huge Limitation can’t find a proper data set to train a model for identifying the components.
2. There has been added only iOS HIG principles for this knowledge base. Therefore, can't detect android violations.
3. And also, can't detect the violations of animations.
4. When inputting the meta jason, input it that is related to the correct component. If it is not so, the outcoming result is wrong. There has created a sample perfect json for related components to the testing and evaluations for now.

10.9 Future Enhancements

Autor will integrate this tool with design tools (such as Figma), and with simiulators (such as with XCode simitators for iOS) for real-time UI violation detection.

CREATEARO will support implicit as well as de-facto android guidelines, iOS animation guidelines, other design a system that describe HIG guidelines for UI components.

Wireframes will be used to implement HIG guidelines in code base.

10.10 Research Contribution to Achievement

This report is just for Android UIs. However, this does not apply to iOS. Because the principles that govern each of its components differ somewhat from those of android. Furthermore, in Material Design, the components used in each personas face are not smooth, making accurate detection difficult without a HIG.

To overcome this gap, a domain contribution might be made.

10.10.1. Contribution to the Software Engineering domain :

This research will be offering a novel way to identify when to use those components principles and guidelines.

10.10.2. Contribution to the BoK :

This research will be used in the domain to the identification of violations from the user interfaces of iOS applications.

10.11 Concluding Remarks

This chapter concludes the research thesis by evaluating whether the research objectives were met, how the author's skills were used by, how the author managed to overcome barriers, the project's limitations, potential future works, and the ability to contribute to the knowledge base. This research concludes with a contribution that could have a significant impact on a domain that could impact the future of automation for many outdoor computer vision systems. The research was carefully planned prior to its initiation and carried out in line with the plan throughout all phases. It received overwhelmingly positive responses from academic and industrial experts with vast experience in this and other given the technological fields.

References

Hegghammer, T., 2022. OCR with Tesseract, Amazon Textract, and Google Document AI: a benchmarking experiment. *Journal of Computational Social Science*, 5(2022), p. 861–882.

Bo Pang, E. N. Y. N. W., September 10, 2019. *Deep Learning With TensorFlow: A Review*. s.l., s.n.

Chirag Patel, A. P. P. D. P., October 2012 . Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study. *International Journal of Computer Applications (0975 – 8887),* 55(2012), p. 10.

Gary Bradski, A. K., Sep 24, 2008. *Learning OpenCV: Computer Vision with the OpenCV Library.* s.l., s.n.

Nosal, E.-M., 14-17 October 2008. *Flood-fill algorithms used for passive acoustic detection and tracking.* Hyeres, France, 2008 New Trends for Environmental Monitoring Using Passive Systems.

S.V.Burtsev, Y., 5, September–October 1993. An efficient flood-filling algorithm. *Computers & Graphics* , Volume 17, pp. 549-561.

Arya Chowdhury Mugdha, F. S. R. M. U. A., n.d. *A study of recursive least squares (RLS) adaptive filter algorithm in noise removal from ECG signals.* Department of Electrical and Electronic Engineering, University of Dhaka, Dhaka, Bangladesh, IEEE.

Cheon, B.-W. K. N.-H., Received : 2019.04.26 Accepted : 2019.05.08 Published : 2019.08.31. *A Filter Algorithm using Noise Component of Image in Mixed Noise Environments.* Dept. of Control and Instrumentation Eng., Pukyong National University, Pukyong National University.

Cheon, B.-W. K. N.-H., 2019.08.31 . *A Filter Algorithm using Noise Component of Image in Mixed Noise Environments.* Dept. of Control and Instrumentation Eng., Pukyong National University, Pukyong National University.

Youngmin Baek, B. L. D. H. S. Y. H. L., 2019. *Character Region Awareness for Text Detection.* s.l., Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

Ratri Dwi Atmaja, M. A. M. J. H. F. Y. S., August 2016. An Image Processing Method to Convert RGB Image into Binary. *Indonesian Journal of Electrical Engineering and Computer Science,* Volume Vol. 3, No. 2, p. 377 ~ 382.

Tarun Kumar, K. V., September 2010. A Theory-Based on Conversion of RGB image to Gray image. *International Journal of Computer Applications (0975 – 8887),* Volume 7, p. 2.

Kari Pulli, A. B. K. K. V. E., 2012. Real-time computer vision with OpenCV. *Communications of the ACM,* Volume 55, Number 6(6), pp. 61-69.

MeiZhang, J. H. P. Y., 2017. The Application of One-Class Classifier Based on CNN in Image Defect Detection. *Procedia Computer Science,* Volume 114, pp. 341-348.

Inc, A., 2022. *Human Interface Guidelines.* [Online] Available at: https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/

Google, n.d. *Material Design.* [Online] Available at: https://material.io/components?platform=ios

Moran, K., May 2018. Automated reporting of GUI design violations for mobile apps.. *ICSE,* Issue May 2018165-175.

Nguyen, T. T., May 2018. Deep learning UI design patterns of mobile apps.. *ICSE-NIER,* pp. 65-68.

Imran, A., Dec 2019. Design Smell Detection and Analysis for Open Source Java Software. *IEEE,* Issue Oct 2019.

Ruiz, J., 2022. Learning UI Functional Design Principles Through Simulation With Feedback. *IEEE - Transactions on Learning Technologies* , 3(2022).

Yang, B., 2021. *Don't Do That! Hunting Down Visual Design Smells in Complex UIs Against Design Guidelines.* s.l., IEEE.

Mulong Xie, S. F. Z. X. J. C. C. C., 2020. *UIED: A Hybrid Tool for GUI Element Detection.*. Canberra, Melbourne, Australian National University, Monash University..

MulongXie, 2021. *UIED.* [Online] Available at: https://github.com/MulongXie/UIED [Accessed 30 11 2021].

Valéria Lelli, A. B. ,. B., June,2016. *Automatic Detection of GUI Design Smells : The Case of Blob Listener*.. s.l., ACM.