



**INFORMATICS
INSTITUTE OF
TECHNOLOGY**

INFORMATICS INSTITUTE OF TECHNOLOGY

In Collaboration with

UNIVERSITY OF WESTMINSTER

Createaro - Automating Violation Detection on iOS UI

A Project Specification Design and Prototype by

Miss. Chavika Kodithuwakku

Supervised by

Mr. Asanjay Fernando

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

February 2022

Table of Contents

CHAPTER 1: INTRODUCTION	1
1.1 Chapter Overview	1
1.2 Problem Domain	1
1.3 Problem Definition.....	3
1.3.1 Problem Statement	3
1.4 Aims and Objectives	3
1.4.1 In-Scope Aims	3
1.4.2 Out-Scope Aims	4
1.5 Novelty Of Research	4
1.6 Research Challenge.....	4
1.7 Chapter Overview	5
CHAPTER 2: SOFTWARE REQUIREMENT SPECIFICATION	5
2.1 Chapter Overview	5
2.2 Rich Picture Diagram.....	5
2.3 Stakeholders Analysis	6
2.3.1 Stakeholder Onion Model	7
2.3.2 Stakeholder Viewpoints	7
2.4 Selection of Requirement Elicitation Method.....	9
2.4.1 Justifications for the Selection of RE Techniques	9
2.4.2 Justifications for the non- Selection of RE Techniques	10
2.5 Discussion of Results	11
2.5.1 Findings from LR.....	11
2.5.2 Questions.....	11
2.5.3 Prototyping.....	15
2.5.4 Self-evaluation	17
2.6 Summery of Findings.....	17

2.7 Context Diagram	18
2.8 Use Case Diagram.....	19
2.9 User Description	19
2.10 Requirements	21
2.10.1 Non- Functional Requirements need to be prioritized using MoSCoW principles.	21
2.10.2 Functional Requirements need to be prioritized using MoSCoW principles.	22
2.11 Chapter Summary	23
CHAPTER 3: SYSTEM ARCHITECTURE AND DESIGN	23
3.1 Chapter Overview	23
3.2 Design Goals	23
3.3 System Architecture Design	24
3.3.1 Tiered Architecture	25
3.4 System Design	27
3.4.1 Choice of the Design Paradigm	28
3.5 Design Daigram	28
3.5.1 Data flow diagram.....	28
3.5.4 UI Design	29
3.5.5 User Experience flow diagram.....	32
3.5.6 System Process Flow Chart	33
3.6 Chapter Summery	34
CHAPTER 4: IMPELMANTATION	34
4.1 Chapter Overview	34
4.2 Technology Selection.....	34
4.2.1 Technology stack	34
4.2.2 Data Selection	35
4.2.3 Development Framework.....	35
4.2.4 Programming Language.....	36
4.2.5 Libraries	36

4.2.6 IDE.....	37
4.2.7 Summary of Technology Selection.....	37
4.3 Implementation of Core Functionalities	38
4.4 Self-Reflection	41
4.5 Video Demo	43
4.6 Chapter Summary	43
References.....	44

Table of Figures

Figure 1 - Rich Picture Diagram.....	6
Figure 2 - Stakeholder Onion Model	7
Figure 3 - Context Diagram	18
Figure 4 - Use Case Diagram.....	19
Figure 5 - Tiered Architecture	25
Figure 6 - Data flow diagram.....	28
Figure 7 - Sequence diagram	29
Figure 8 - UI Design	31
Figure 9 - User flow	32
Figure 10 - System Process Flow Chart.....	33
Figure 11 - Technology stack	35
Figure 12 - view.py code	39
Figure 13 - runner.py code.....	39
Figure 14 - models.py code.....	40
Figure 15 - Self-Reflection	41

Table of Tables

Table 1 - Stakeholder Viewpoints.....	9
Table 2 - Justifications for the Selection of RE Techniques.....	10
Table 3 - Justifications for the non- Selection of RE Techniques	11
Table 4 - Findings from LR	11
Table 5 - Questions	15
Table 6 - Prototyping	17

Table 7 - Summery of Findings	18
Table 8 - User Description - Input application screen	20
Table 9 - User Description - Manual annotation	20
Table 10 - Non- Functional Requirements.....	22
Table 11 - Functional Requirements	23
Table 12 - Design Goals	24
Table 13 - Tiered Architecture.....	27
Table 14 - Libraries.....	37
Table 15 - Summary of Technology Selection	38
Table 16 - Implementation of Core Functionalities	38

CHAPTER 1: INTRODUCTION

1.1 Chapter Overview

In this proposal, a tool that can auto detect UI violations and human interface guidelines in line with User Interface(UI) and Human Interface Guidelines (HIG) principles is discussed. The existing works build access to the research gap. And the research challenges are generalizing the research.

1.2 Problem Domain

When developing an application, it is essential to focus on both the functional development as well as the UI.

The abbreviation UI is used for the term 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.

Suppose the person is a developer, a designer, or a content developer. 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 circumstances, 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 for 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, 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 and their aspects and various dimensions, providing an application to a user which gives the best experience to 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 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 about the field of application development. Consequently, many people release new applications.

But due to the violation of UI principles and Human Computer Interaction (HCI) principles from these applications, there is a considerable potential of 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 his or her purpose.

If there is no good connection between the user and functionalities, the application may be considered as 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 iOS application, the user base experiences a bad experience, and the users tend to uninstall apps.

1.4 Aims and Objectives

1.4.1 In-Scope Aims

1. Through this paper, details that are relevant for each and every component of iOS Guidelines of HIG are mentioned in detail.
2. The project is about the demographic studies regarding Material Design (MD). 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.

1.4.2 Out-Scope Aims

1. Employing Implicit and de-facto guidelines in guidelines.
2. Utilizing development tools 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.

1.5 Novelty Of Research

The following are the key aims of this research study.

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

1.6 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

They only differ from how they are described in HIG. There are not any kind of differences in component design aspects, general design dimensions, and atomic UI information involved (Bo Yang, 2020).

Therefore, implicit design guidelines, which have the potential for supporting in the same way as the explicit guidelines once they have been discovered (Bo Yang, 2020).

Depending on the official iOS guidelines, it is possible to derive certain principles from the 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 of such a flow via an icon. But there might be incidents where a user can show feedback of 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.

1.7 Chapter Overview

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 a snag.

CHAPTER 2: SOFTWARE REQUIREMENT SPECIFICATION

2.1 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.

2.2 Rich Picture Diagram

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

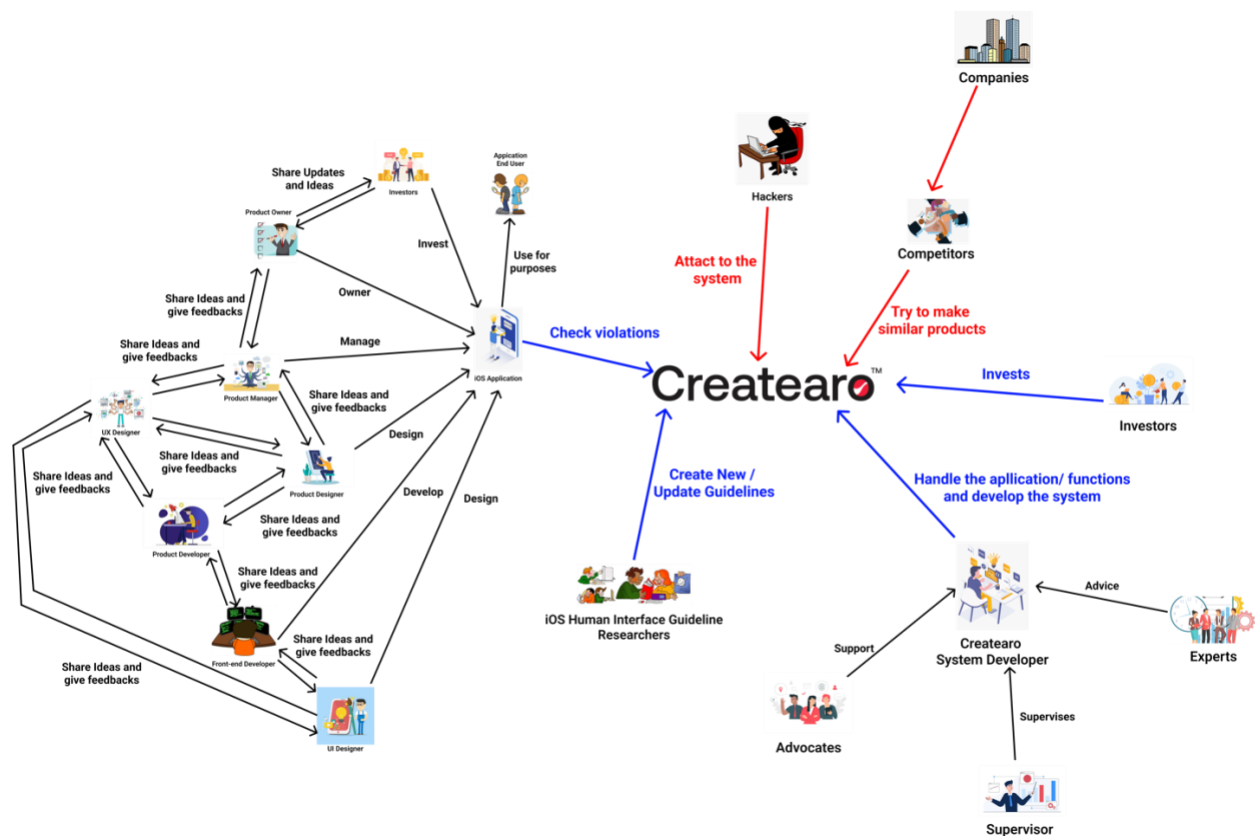


Figure 1 - Rich Picture Diagram

2.3 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.

2.3.1 Stakeholder Onion Model

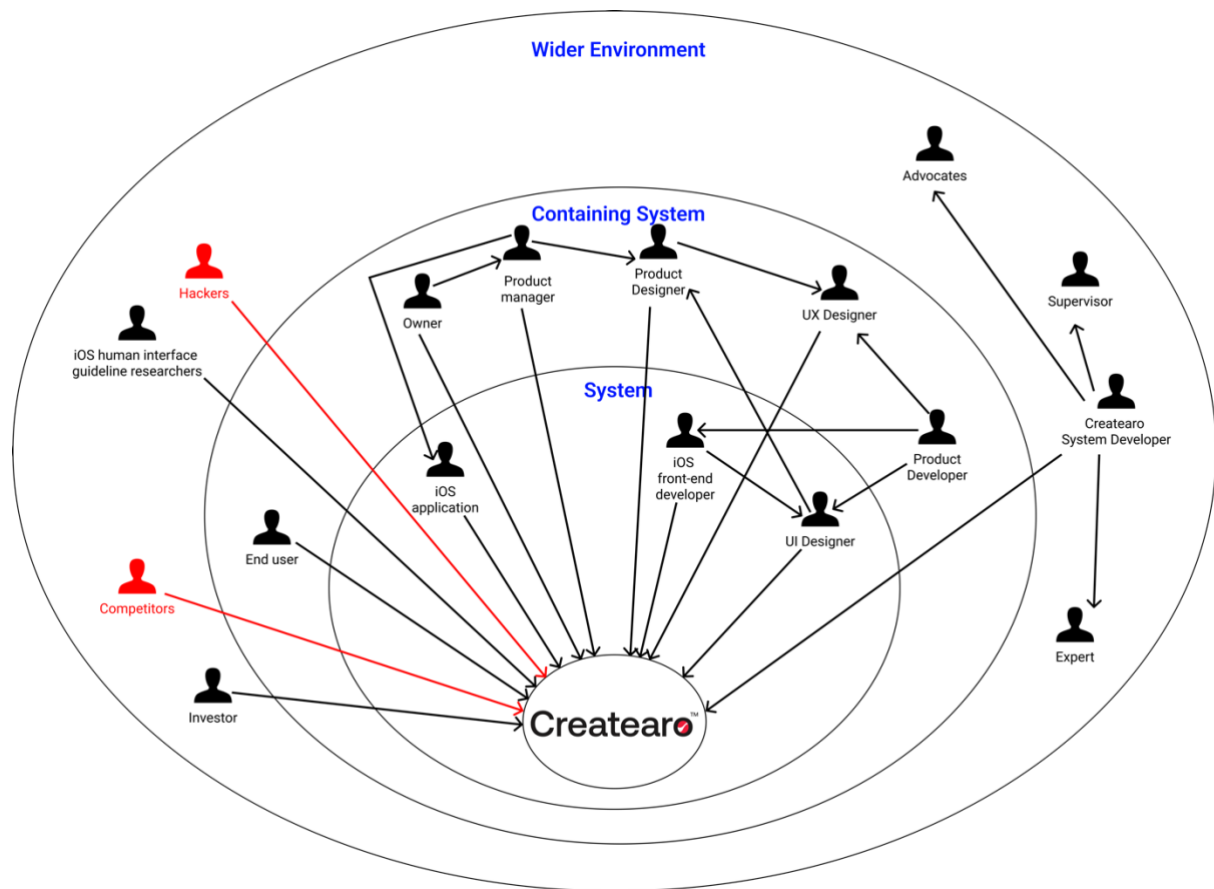


Figure 2 - Stakeholder Onion Model

2.3.2 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/ product manager/ product 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 stakeholder	Hackers may attempt to modify the system's important input images in some other way, or they may just try to create damage with the system as a whole.
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 stakeholder	Try to develop similar products.
iOS human interface guideline researchers	Domain Expert	Develop new rules or update rules.

Table 1 - Stakeholder Viewpoints

2.4 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.

2.4.1 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 a 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.

<p>When it comes time to finalizing 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.</p> <p>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.</p>
3. Questionnaire
<p>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.</p>
4. Self-Evaluation
<p>Self-assessment was carried out at several stages of the project, including scope definition and refinement, the identification of new needs, and the rating and ranking of requirements. There has used a Tesseract OCR for text detection part and pure computer version-based model for components detection part. Here if you have detected the component from the component detection model, and also if there is a test it will not be detected. Sometimes non components are also detected. Therefore, so researched another algorithm.</p> <p>Have an idea to use the EAST algorithm instead of Tesseract OCR for text detection. Model that has used binarization and block detection.</p>
5. Prototyping
<p>Using the "Prototyping" software development process, the project is now underway. Because of this, it is viable to employ prototype as a requirements definition approach because the stages of development are recursive with rapid designing process at the beginning. All through the prototype development process, it will help identify areas for improvement and criteria that must be met.</p>

Table 2 - Justifications for the Selection of RE Techniques

2.4.2 Justifications for the non- Selection of RE Techniques

1. Requirement Workshops
<p>The ability to bring together a diverse collection of stakeholders for a requirement workshop is a major drawback.</p>
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 3 - Justifications for the non- Selection of RE Techniques

2.5 Discussion of Results

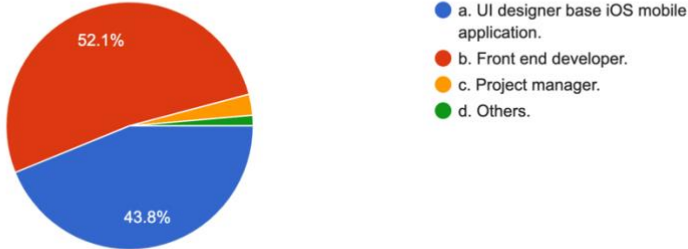
2.5.1 Findings from LR

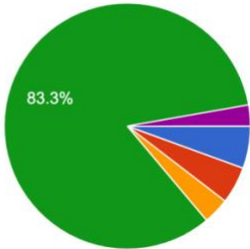
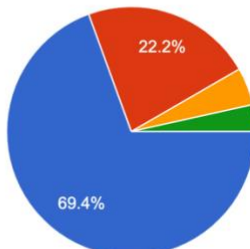
Findings	Citation
<p>Component and text detect interpreted.</p> <p>When it comes to the design of user interfaces, one of the most common technique for detecting elements is the use of computer vision (CV).</p> <p>For example, a mobile app image might be used as an input for UI Element Detection (UIED), 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.</p> <p>UI text and visual components, such as buttons, images, and input bars, are detected by UIED using a two-part architecture.</p> <p>Detection of text is handled using Google OCR.</p> <p>To find and classify graphical components, it makes use of traditional CV techniques and a CNN classifier.</p>	<p>(Mulong Xie, 2020) (MulongXie, 2021)</p> <p>(Valéria Lelli, June 2016)</p>

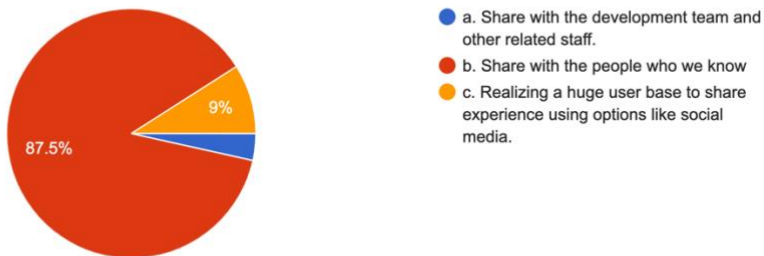
Table 4 - Findings from LR

2.5.2 Questions

Question	Choose your passionate carrier,
– no.1	<ul style="list-style-type: none"> a. UI designer base iOS mobile application. b. Front end developer.

	<p>c. Project manager.</p> <p>d. Others.</p>
Aim	User identification and filtering.
Findings	<p>146 responses</p>  <p>As it was expected, almost all the participants (more than 95%) of the survey will be a front-end developer or a UI designer.</p> <p>From here we can associate that the people who were involved in this survey have knowledge about this.</p> <p>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."</p>
Question – no.2	<p>How would you describe yourself in the domain of UI principles?</p> <p>a. Beginner.</p> <p>b. I'm an expert in iOS design. I have done so much designing work. When doing them, I always followed the guidelines.</p> <p>c. I'm an expert in iOS development. If there is an issue with the testing, I have the knowledge to solve it also.</p> <p>d. Just copy and paste the design screens and prototypes into the code. The principle doesn't bother me.</p> <p>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.</p>
Aim	To get an idea about what kind of knowledge that the survey users have about principles.
Findings	

	<p>144 responses</p>  <ul style="list-style-type: none"> ● a. Beginner. ● b. I'm an expert in iOS design. I have done so much designing work. When doing them, I always followed the guid... ● c. I'm an expert in iOS development. If there is an issue with the testing, I hav... ● d. Just copy and paste the design screens and prototypes into the code.... ● e. I'm good at knowledge. I have a good knowledge base on UI principles. Bec... <p>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.</p>
Question – no 3	<p>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,</p> <ul style="list-style-type: none"> 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	<p>To identify what kind of changes are coming. You have to find out that there is a user base for this violation detection tool.</p>
Findings	<p>144 responses</p>  <ul style="list-style-type: none"> ● 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.

	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	<p>144 responses</p>  <p>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.</p>
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	

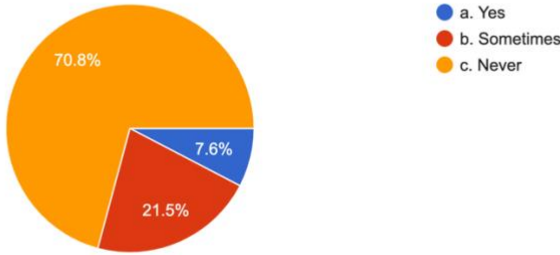
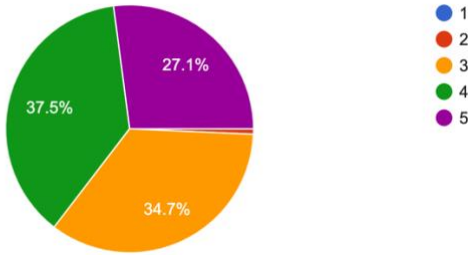
	<p>144 responses</p>  <p>There are two sections of violation. One is a UI that doesn't have responsiveness and the other one that doesn't care about the UI principles. From this survey we can identify that most of the user base don't care about the principles</p>
Question – no.6	<p>How successful would it be if given the chance to detect violation by inputting an image and metadata file/ hierarchy json file (optional)on the application screen?</p> <p>1- Very Poor. 5- Excellent.</p>
Aim	<p>Research into establishing whether or not professionals in the relevant field are examining the suggested solution is important.</p>
Findings	<p>144 responses</p>  <p>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.</p>

Table 5 - Questions

2.5.3 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	In order to see how well the model will perform with

	should only be applied to a specific domain at a moment.	various hyperparameter values, "apply for a given domain." It would better. If the modules can be connected with other general domains
--	----------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------

Table 6 - Prototyping

2.5.4 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.

2.6 Summery of Findings

Findings	Literature Review	Formal Interviews	Questionnaire	Self-evaluation	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 7 - Summary of Findings

2.7 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.

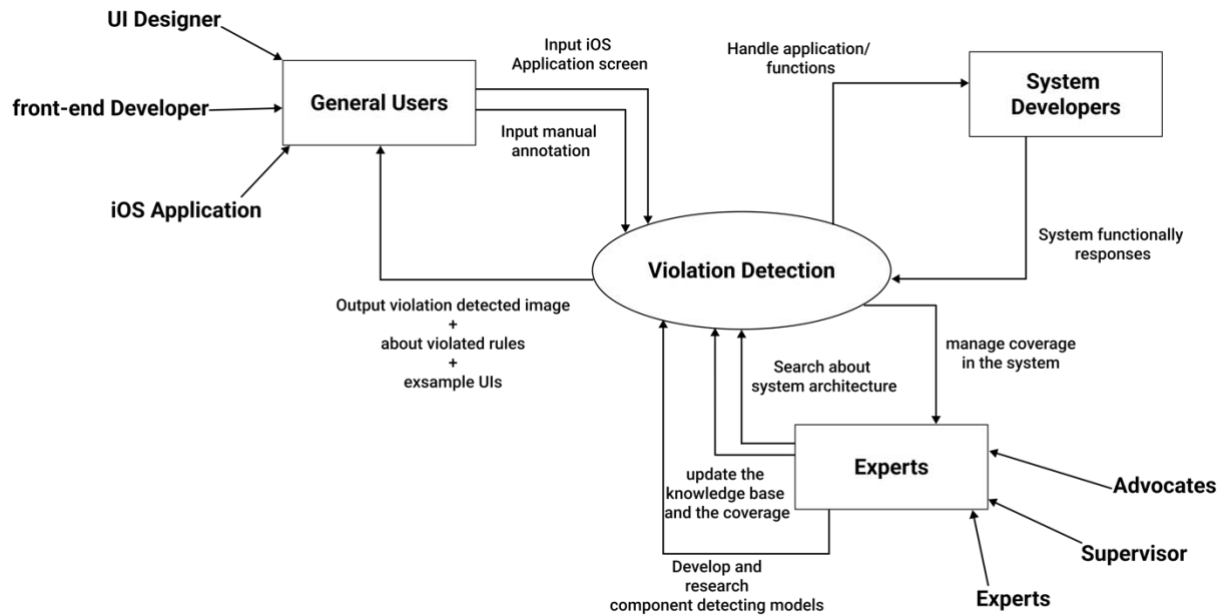


Figure 3 - Context Diagram

2.8 Use Case Diagram

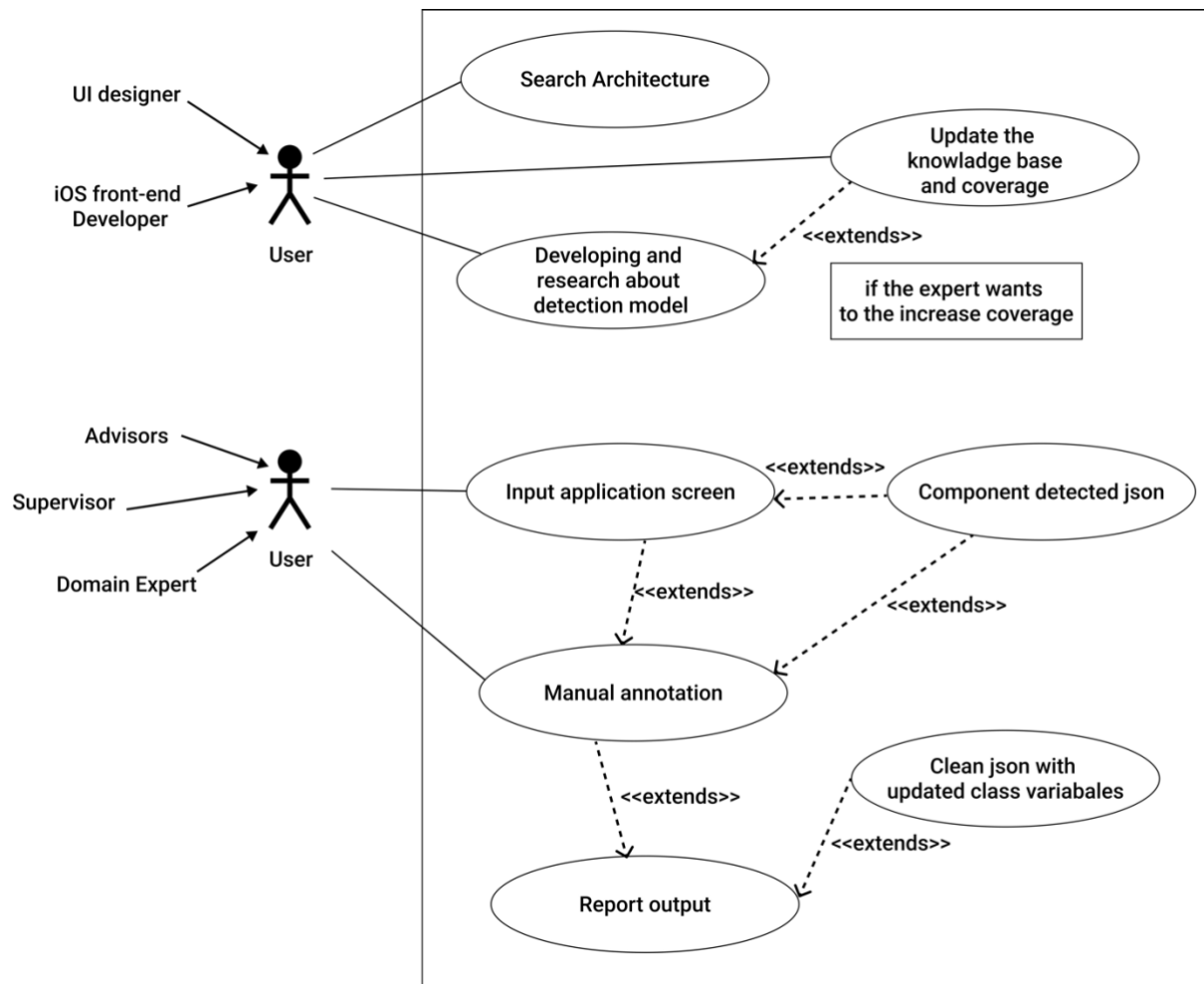


Figure 4 - Use Case Diagram

2.9 User Description

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	User
Pre-conditions	In order to identify infractions, the user must possess a picture.
Extended use cases	Display an Error Message and an uploaded Image.
Included use cases	None

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

Table 8 - User Description - Input application screen

Use Cases Name	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 9 - User Description - Manual annotation

2.10 Requirements

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

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.

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

ID	Requirement	Description	Priority Level
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. For the first time user, the application should have a low learning curve, making it easy for them to get up and running quickly. In order to keep people informed during the entire process, it is essential to keep them up to date.	C
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

Table 10 - Non- Functional Requirements

2.10.2 Functional Requirements need to be prioritized using MoSCoW principles.

ID	Requirement Description	Use case mapping	Priority Level
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 11 - Functional Requirements

2.11 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 3: SYSTEM ARCHITECTURE AND DESIGN

3.1 Chapter Overview

This chapter goes deeply into the design part of the project. It extends from the system's core to the user interface.

Design selections were made based on requirements gathered through a literature research, in-depth interviews, and a questionnaire, as well as findings from the prototyping process.

Additionally, the rationale behind certain design decisions is explored here.

3.2 Design Goals

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

Design Goal	Description
Ease to use	<p>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.</p> <p>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.</p>

Performance and Reduce Latency	<p>Python is used for the back end. Django is also a Python framework. Because of this, latency is reducing.</p> <p>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.</p>
Reusability	<p>It is important for the modules of the system to be created in a way that allows them to be changed.</p> <p>This is especially important for research that helps other researchers in the field.</p> <p>If anyone can change a part of the system with another, they can try some new things or improve the system.</p> <p>Example they can add new components/ design aspects/ design dimensions according to the future trends.</p>
Correctness	<p>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.</p>
Scalability	<p>The system should be able to use large datasets for training because there will be more datasets for ios apps in the future.</p> <p>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.</p> <p>Also, the system must be able to run on a server that can serve a lot of people at the same time.</p>

Table 12 - Design Goals

3.3 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.

3.3.1 Tiered Architecture

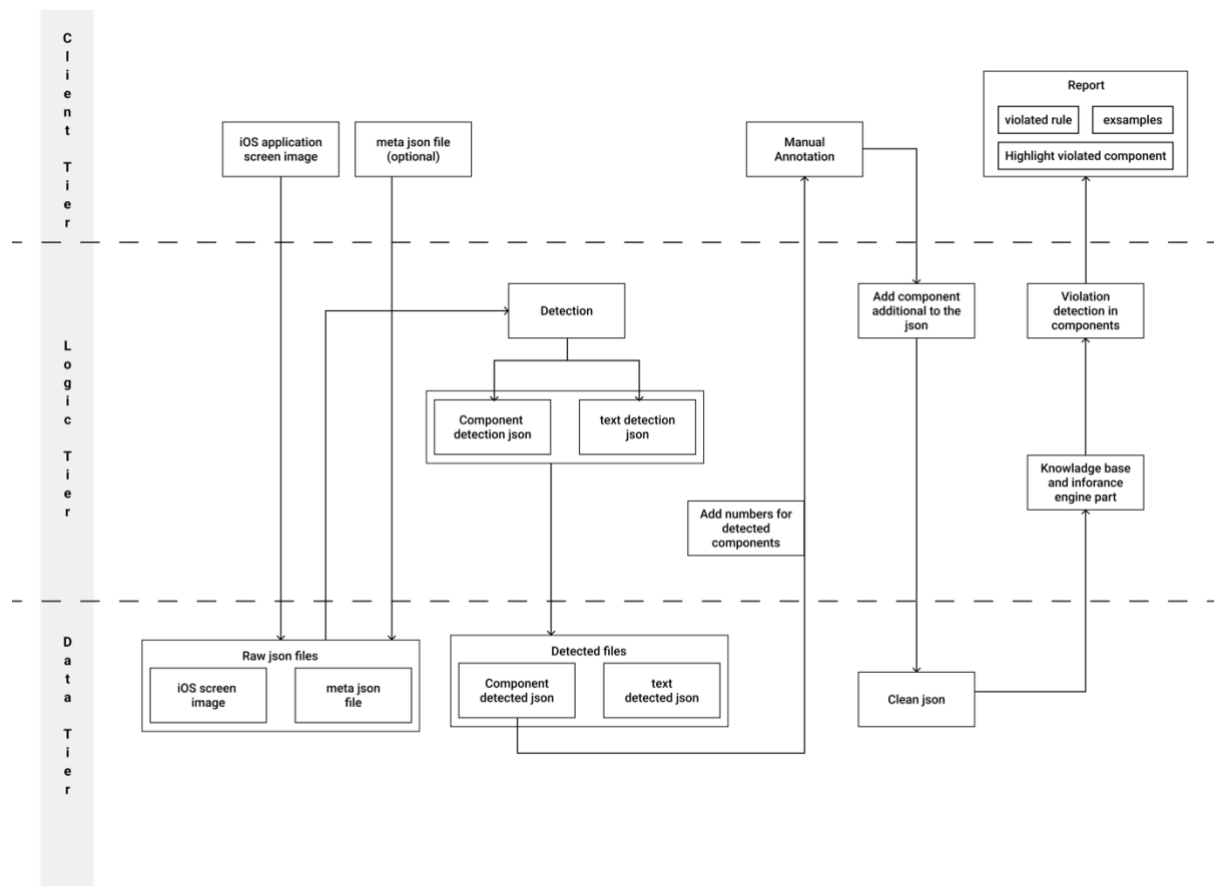


Figure 5 - Tiered Architecture

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 more simple and 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	<p>This is the final output. It comes from e categories. One is highlighting the place that violation has placed.</p> <p>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.</p>
Logic tier	
Detection	<p>Here two main detections are happening.</p> <ol style="list-style-type: none"> 1. Text detection- Has used Tesseract OCR. 2. Component detection- pure computer vision base technique. <p>Here the detection is based on the screen that the user has input at the client tier.</p> <p>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.</p> <p>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.</p>

	<p>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.</p> <p>Adding a numberrign for all the components that has come from the 1st erection is the thing that is happening in this part.</p> <p>Because it is user friendly for the user in the manual annotation.</p> <p>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.</p>
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 jsonal 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 13 - Tiered Architecture

3.4 System Design

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

3.4.1 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.

3.5 Design Daigram

3.5.1 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.

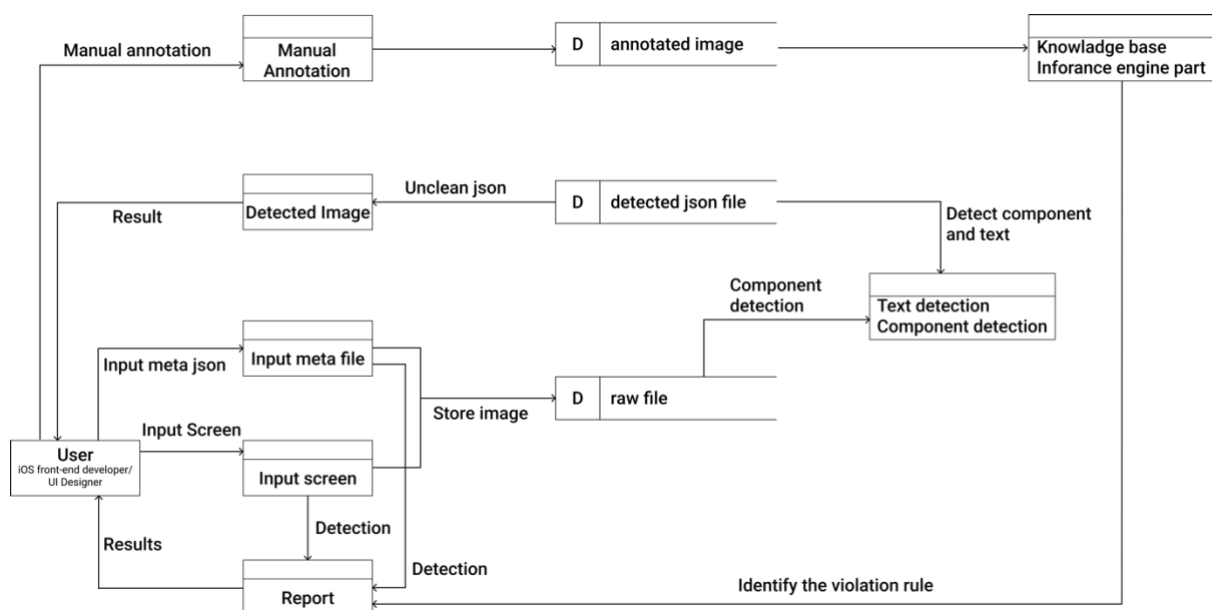


Figure 6 - Data flow diagram

3.5.2 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.

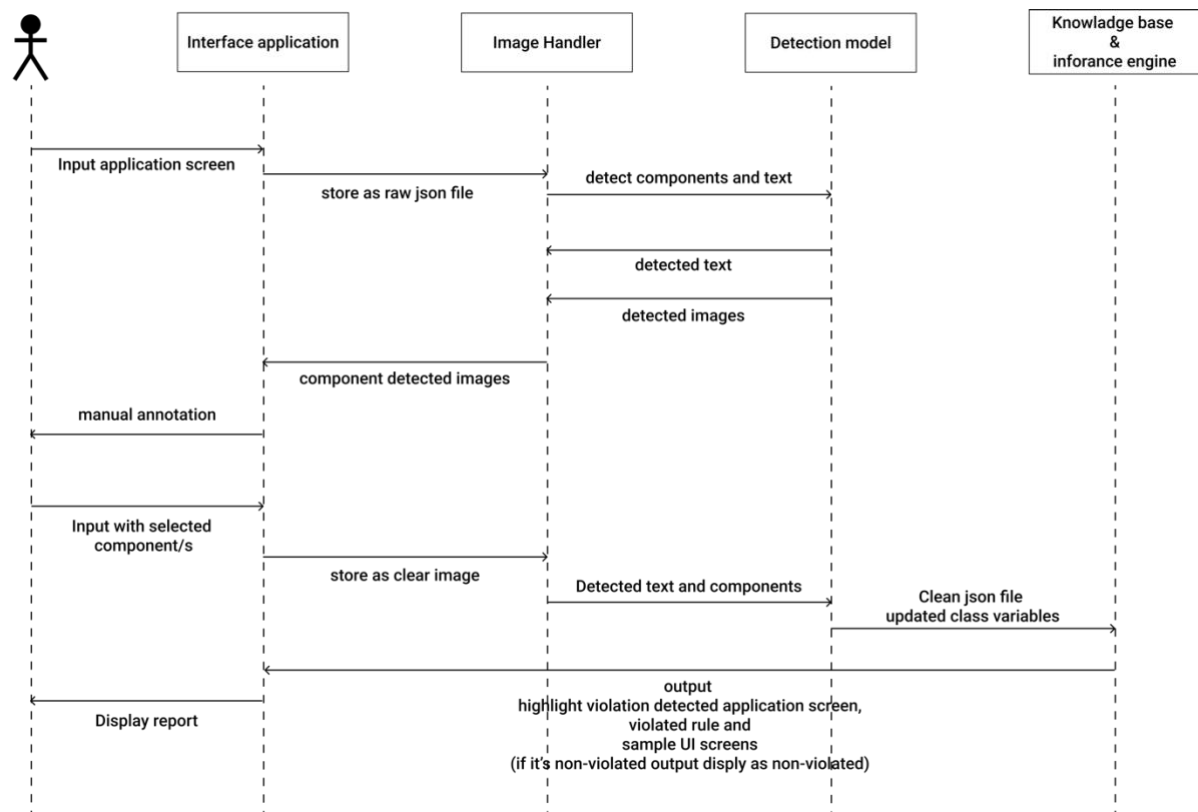


Figure 7 - Sequence diagram

3.5.3 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.

3.5.4 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.

Createaro™

DESIGN GUIDELINE GALLERY

VIOLATIONS DETECTION

Bottom Navigation ▾ Typography ▾ Select Design Aspects ▲ Severity of violations ▾ Search components

Usages
Anatomy
Behavior
Placement

NO.1 - bottom-bar-min-button
Class - Layout - Usage - **Warning**

Content -
Limit the number of buttons on a bottom bar for more than one.
(floating action button doesn't count).

Needed Information - Metadata Extractor

Violations sample UI

NO.1 - bottom-bar-min-button
Class - Layout - Usage - **Warning**

Content -
Limit the number of buttons on a bottom bar for more than one.
(floating action button doesn't count).

Needed Information - Metadata Extractor

Conformance sample UI

Violations sample UI

Createaro™

DESIGN GUIDELINE GALLERY

VIOLATIONS DETECTION

Bottom Navigation ▾ Typography ▾ Select Design Aspects ▲ Severity of violations ▾ Search components

Warning
Permission
Error

NO.1 - bottom-bar-min-button
Class - Layout - Usage - **Warning**

Content -
Limit the number of buttons on a bottom bar for more than one.
(floating action button doesn't count).

Needed Information - Metadata Extractor

Conformance sample UI

Violations sample UI

NO.1 - bottom-bar-min-button
Class - Layout - Usage - **Warning**

Content -
Limit the number of buttons on a bottom bar for more than one.
(floating action button doesn't count).

Needed Information - Metadata Extractor

Conformance sample UI

Violations sample UI

NO.1 - bottom-bar-min-button
Class - Layout - Usage - Warning
Content -
 Limit the number of buttons on a bottom bar for more than one.
 (floating action button doesn't count).
Needed Information - Metadata Extractor

Conformance sample UI

Hurray, We found jailbreak tools for iPhone 11 Pro Max iOS 14.3

What's Next?

Get ahead and complete the next few steps to get all of your favorite apps including YouTube on jailbreak.

[Find Available Jailbreak Tools](#)

Violations sample UI

Search For Categories

Main Categories

Apple iOS

Other

Warning

This release is in beta preview and as such should not be installed on a primary device.

[Download TweakMe](#)

Highlighted changes

Available only on Apple TV OS during normal system operation.

Bug fixes

Fixes A12 and A13 devices crashing when trying to jailbreak on iOS 14.3 or higher.

Fixes an issue where some LaunchDaemons were installed on platforms where they should not have been.

Createaro
DESIGN GUIDELINE GALLERY
VIOLATIONS DETECTION

Bottom Navigation

- App Bar - Bottom
- App Bar - Top
- Bottom Navigation
- Buttons
- Buttons - Floating Action
- Lists
- Cards
- Navigation Drawer
- Text Fields
- Dialogs
- Banners

Select Design Dimentions

Select Design Aspects

Severity of violations

Serach components

Conformance sample UI

Violations sample UI

NQ.1 - bottom-bar-min-button

Class - Layout - Usage - Warning

Content -

Limit the number of buttons on a bottom bar for more than one.
(floating action button doesn't count).

Needed Information - Metadata Extractor

Conformance sample UI

Violations sample UI

Figure 8 - UI Design

3.5.5 User Experience flow diagram

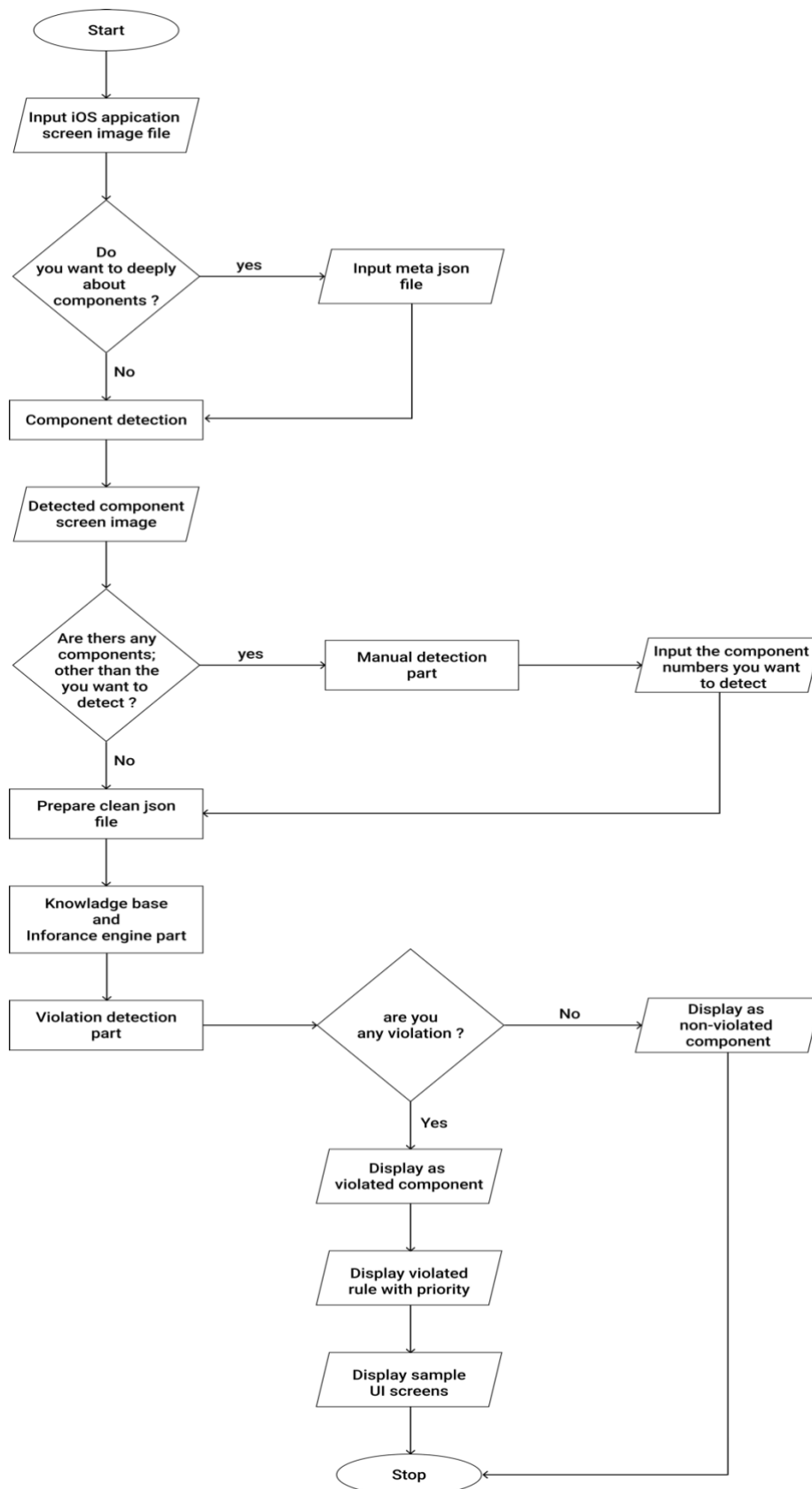


Figure 9 - User flow

3.5.6 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.

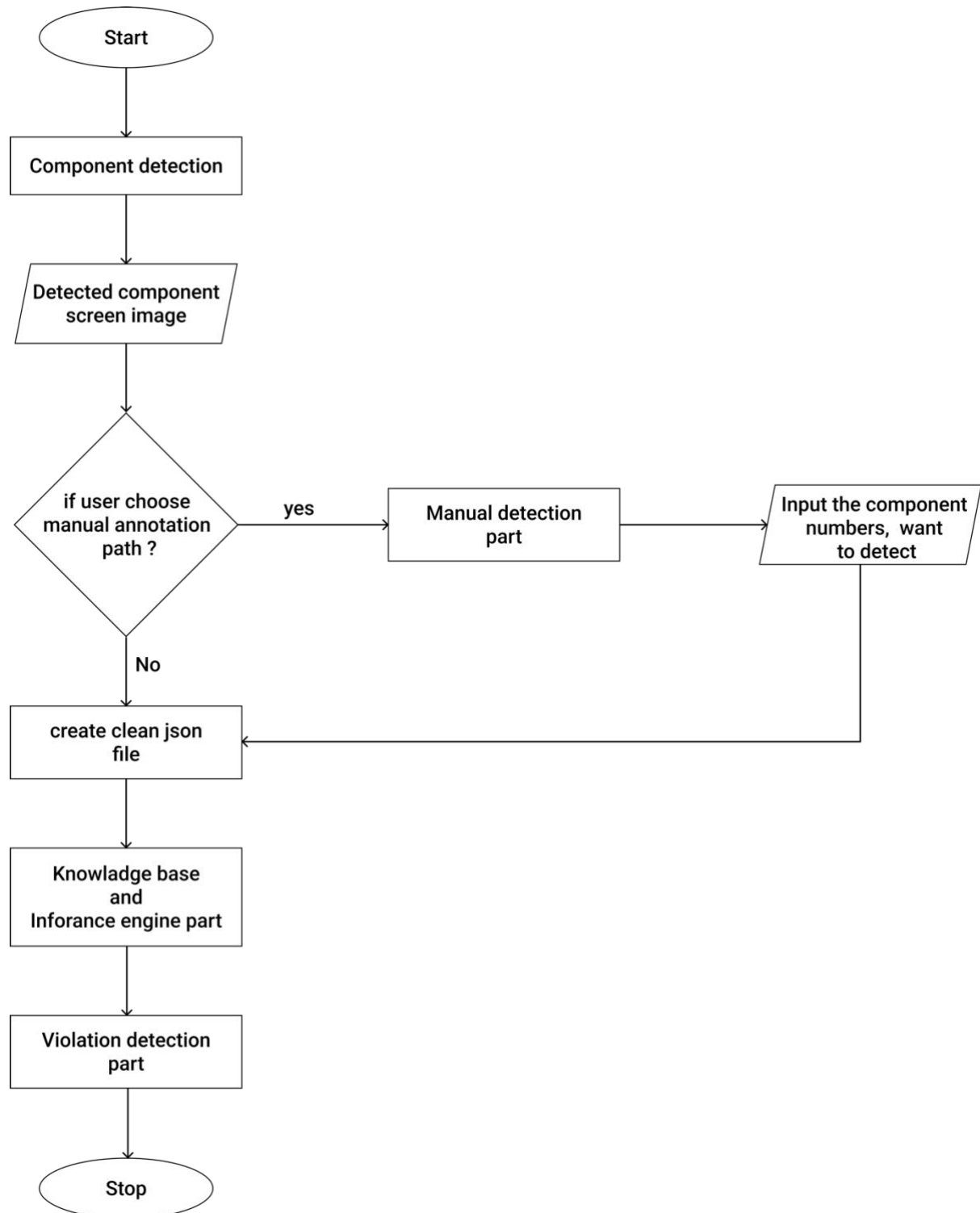


Figure 10 - System Process Flow Chart

3.6 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 modularised 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 4: IMPEMANTATION

4.1 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.

4.2 Technology Selection

4.2.1 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 we 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.














Back-end Stack	IDEs	Libraries
 	 	   
	APIs	
		
Front-end Stack	Version Control	
 	 	

Figure 11 - Technology stack

4.2.2 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.

It is not detected by a trained model that has selected data yet.

4.2.3 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.

4.2.4 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.

4.2.5 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.	<ol style="list-style-type: none"> 1. Open source. 2. Speed. 3. Integration is simple. 4. Programming easy. 5. Prototyping in record speed.
PIL	A different range of image file formats can be opened, edited, and saved.	<ol style="list-style-type: none"> 1. The modification of each pixel location. 2. Masking and transparency management. 3. Sharpening, adjusting the brightness, contrast, or color of an image.
json	Commonly used in web applications for sending data.	<ol style="list-style-type: none"> 1. Light in weight.

		<ol style="list-style-type: none"> 2. Text-based data transmission standard that can be understood by humans. 3. Simple format to deal with.
NumPy	Working with these arrays objects and procedures for processing these arrays are stored in a library.	<ol style="list-style-type: none"> 1. N-dimensional array object with high performance. 2. Data can be stored in a multidimensional container here.
Pandas	Built in Python for the purpose of data analysis and manipulation.	<ol style="list-style-type: none"> 1. 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.	<ol style="list-style-type: none"> 1. Because they shouldn't have to start from scratch, the developers save a lot of time this way.

Table 14 - Libraries

4.2.6 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.

4.2.7 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 15 - Summary of Technology Selection

4.3 Implementation of Core Functionalities

File structure of the component detection consists of 3 main parts.

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 files that want to detect the components.

Table 16 - Implementation of Core Functionalities

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.

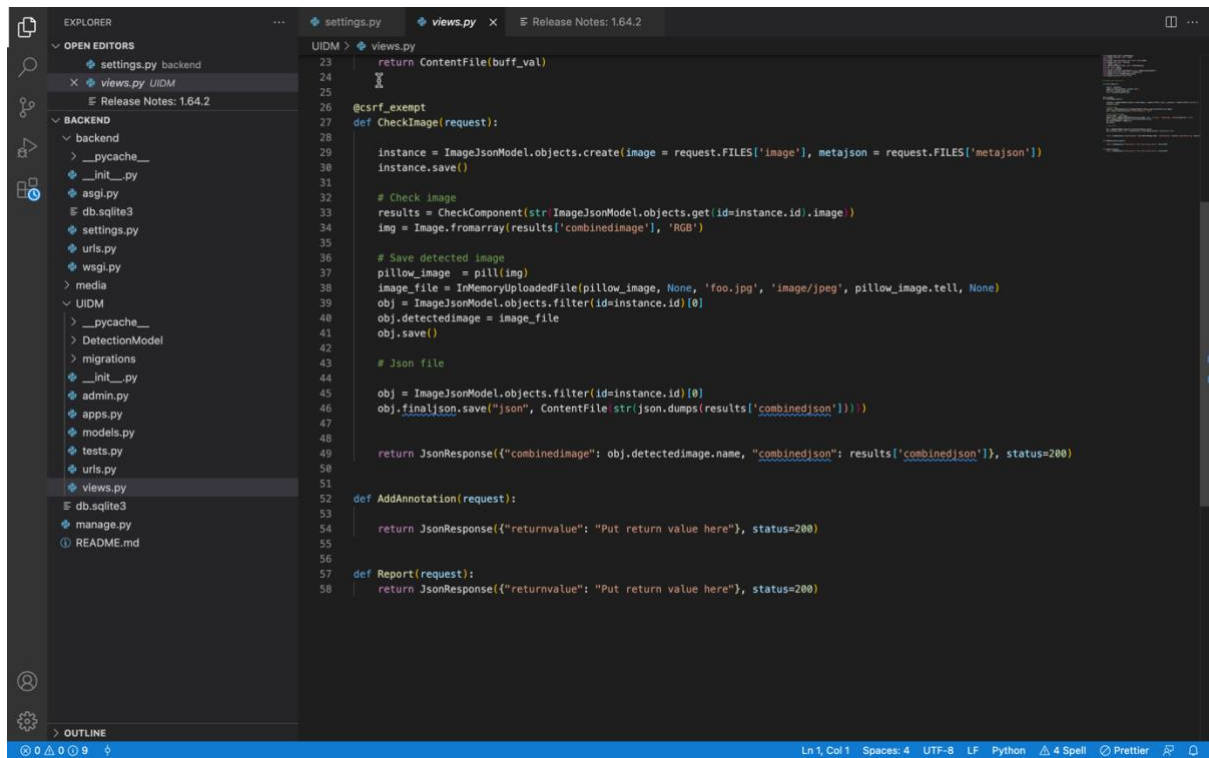


Figure 12 - view.py code

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.

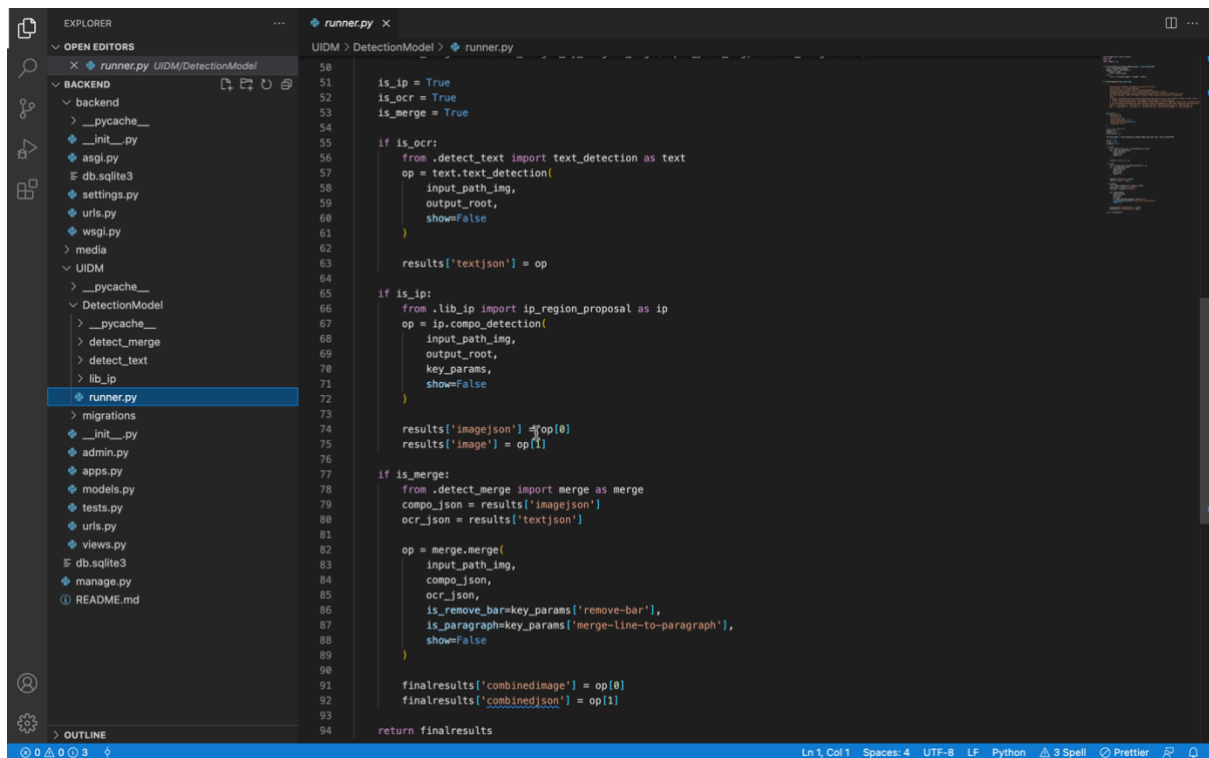


Figure 13 - runner.py code

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

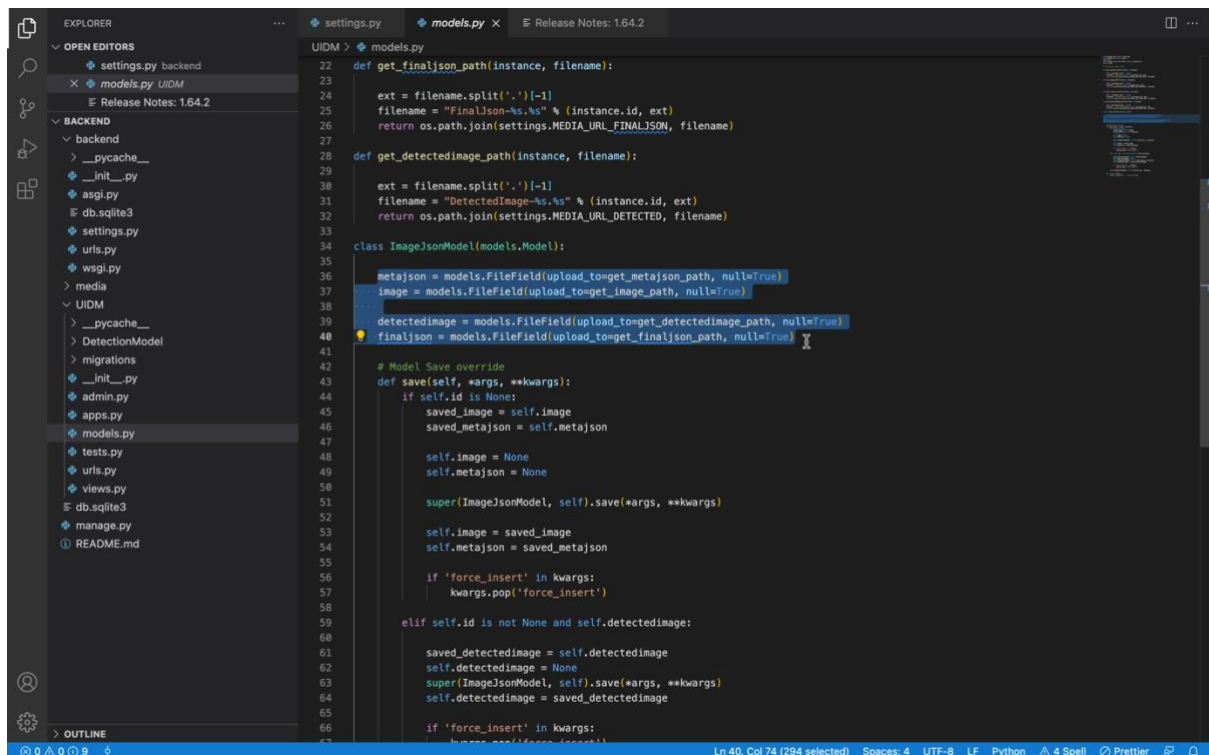


Figure 14 - models.py code

There is 4 columns in the database of model.py

1. Meta Json column- once the metajson 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.

4.4 Self-Reflection

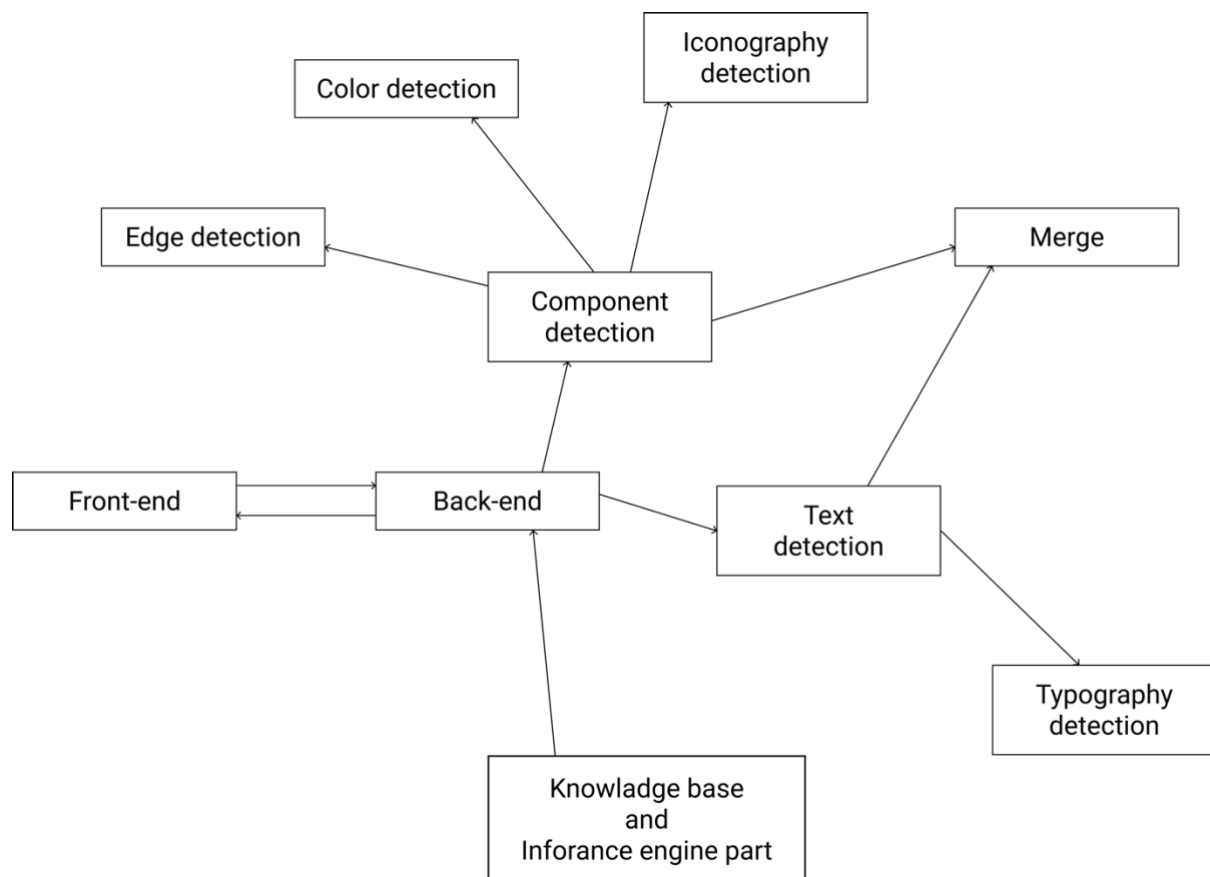


Figure 15 - Self-Reflection

Front End part is the one which interacts with the user. As called user interface.

All the detection part happens in the backend part.

There are two main parts of this detection.

1. Component detection part
2. Text detection part

After that the json that has come after merging this should send again to the front end, to make a manual annotation by the user. Then again it is sent to the knowledge base and inference engine, and then displays the final report to the user.

From the front end side, there is a rich building of total user interface by basing React at the final submission. Also there is a user guide as a help for the user.

Most of the time, at the backend side, logic that is based on functions has converted to the class based logics. For Now the backend structure is based on function. There is an idea to fully convert it to OOP.

In the component detection, normally there happens edge detection, color detection and iconography detection. Here, Iconography and color detection are not happening through classification models for now. It just happens through image processing. For now there is not an iconography identification, just a detection. Simply, it seems that there is an icon only. Still can't recognize what that icon is. But in the final product, train an iconography model for iconography detection and do the iconography detection through it. The Color detection part is not totally completed. Should train a neural network classification for it. Edge detection is happening successfully with the current solution., but hope to improve it more.

Furthermore, there is a hope to use the EAST algorithm for the typography detection model. The Tesseract algorithm is the one which is used now. But EAST is performing better than tesseract according to a research paper. So, hope to use that algorithm.

If mentioned overall, read more research papers to find out more detection techniques. Also used new methodologies for component detection and tried to combine the new approaches. It means there is only about component detection in some researchers and some are only about text detection. So there is a combination of these two in the detection part as the result of this approach. Have to do more research and various combinations to find which gives the best promising result.

According to the knowledge base side, functions that relate to components which have been detected by manual annotation are called manually. Should do research to make auto execute the related rules while passing the relatable variables.

Hope to bring an alternative solution for it because of not using the Pyke.

Example -: imagine that there is an input screen, screen with a keyboard.

Then we can automatically ignore the component named app bottom bar.

Should improve the knowledge base like that.

4.5 Video Demo

Here is the demo video for PSPD. [Click here.](#)

4.6 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.

References

Bo Yang, Z. X. X. C. Y. L., 2020. Don't Do That! Hunting Down Visual Design Smells in Complex UIs against Design Guidelines.. Hangzhou, China, Zhejiang University.

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. MulongXie/UIED. [Online]

Available at: <https://github.com/MulongXie/UIED>

[Accessed 30 11 2021].

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