CodeZillas

Look It
Software Architecture Document
Face Classification System
Version 1.0

Revision History

Date	Version	Description	Author
22/Feb/19	1.0	Document created using initial plans	S. Sabesan

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

The purpose of this software architecture document is to present a detailed architectural description of the Look It Face Classification System. This document will briefly explain about the use-case view, logical view, process view, implementation view and deployment of the system. This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

The scope of this document is to provide a clear description about the architectural development of the system. This will briefly explain about the use-case view of the users and system, logical view of the internal system using diagrams, process view, implementation view of the system and deployment view at the final stage.
This document is created based on the "4+1" model view of architecture in order to depict the software as accurately as possible.

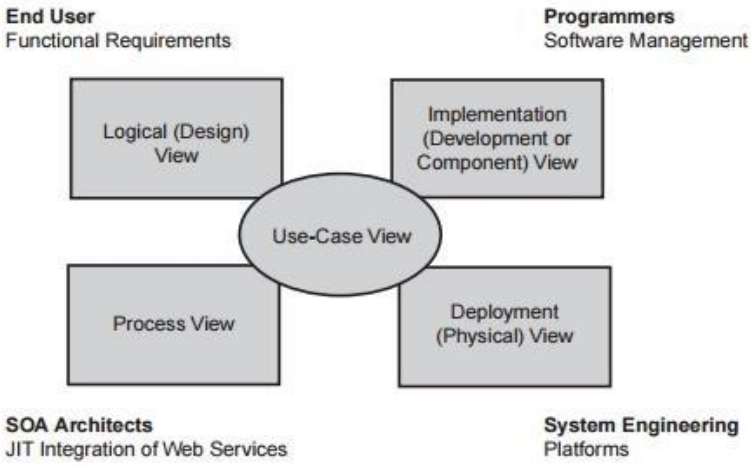


Figure 1 - "4+1" view model

1.3 Definitions, Acronyms, and Abbreviations

- Emotion API – The API which gives results by getting real time pictures from users as input and giving **probability** of an emotion to the given input as output
- User – Person who interacts with the system
- JIT – Just In Time
- SOA – Service Oriented Architecture
- RUP – Rational Unified Process
- MVC – Model View Controller

1.4 References

- "Mapping between 4+1 architectural view model & UML", Software Engineering Stack Exchange, 2019. [Online]. Available: https://softwareengineering.stackexchange.com/questions/233257/mapping-between-41-architectural-view-model-uml.

Tool used to draw the diagrams:

- draw.io. [Online]. Available: www.draw.io
- lucidchart. [Online]. Available: www.lucidchart.com

1.5 Overview

The document explains the design of Look It Face Classification System and provides a clear understanding about the system. The document is divided into several sections to explain the system clearly.

- Architectural representation of the system
- Architectural Goals and Constraints
- Use case view
- Logical view
- Process view
- Deployment view
- Implementation view
- Size and Performance of the system
- Quality of the system

2. Architectural Representation

The Software Architecture document is developed using the views that are defined in the “4+1” model but using the RUP (Rational Unified Process) naming convention. The views used to document the system are:

- Logical view: It describes the functionality of the system that are provided to the users. It can be described by using class diagram or sequence diagram.

- Process view: It explains the system processes and how they communicate. It can be described by using activity diagram.

- Deployment view: It is related to the topology of software components in the physical layer, as well as the physical connection between these components. It can be described using deployment diagram.

- Implementation view: It is related to the software management. It can be described using component diagram or package diagram.

- Use case view: It shows the details of use cases of the system. Moreover, it describes the services provided by the system to users. It can be described using use case diagram.

3. Architectural Goals and Constraints

This section describes the software requirements and objectives that have some significant impact on the architecture.

3.1 Security

As the system deals with Human Face which may be confidential, the system must be secure enough to protect that data. For this face doesn't keep in the system no longer such that avoid from illegal use.

3.2 Persistence

Data persistence of the system will be addressed using a database.

3.3 Performance

As this is an online system, the speed of response of the system is dependent on the internet connection of the users. If the internet connection is good, the users can get quick response from the system.

3.4 Reliability

The availability of the system is very important. The architecture of the system must ensure that the system will be capable enough to avoid system failures. Targeted availability of the system is 24/7.

3.5 Privacy

The result should not be shared with any others than the respective user because they may contain correct information.

3.6 Portability

Since it is an Android based application it can be accessed from anywhere, anytime using a Android device with internet connection.

3.7 Technical platform

As this is an Android based application, the users can easily use the system by entering the online server of the system using this application on their Android mobile phone or tablet. They will require the internet connection to access the online server.

4. Use-Case View

The use case view of the system shows all the use cases in the system which represent some of the important functionalities of the system.

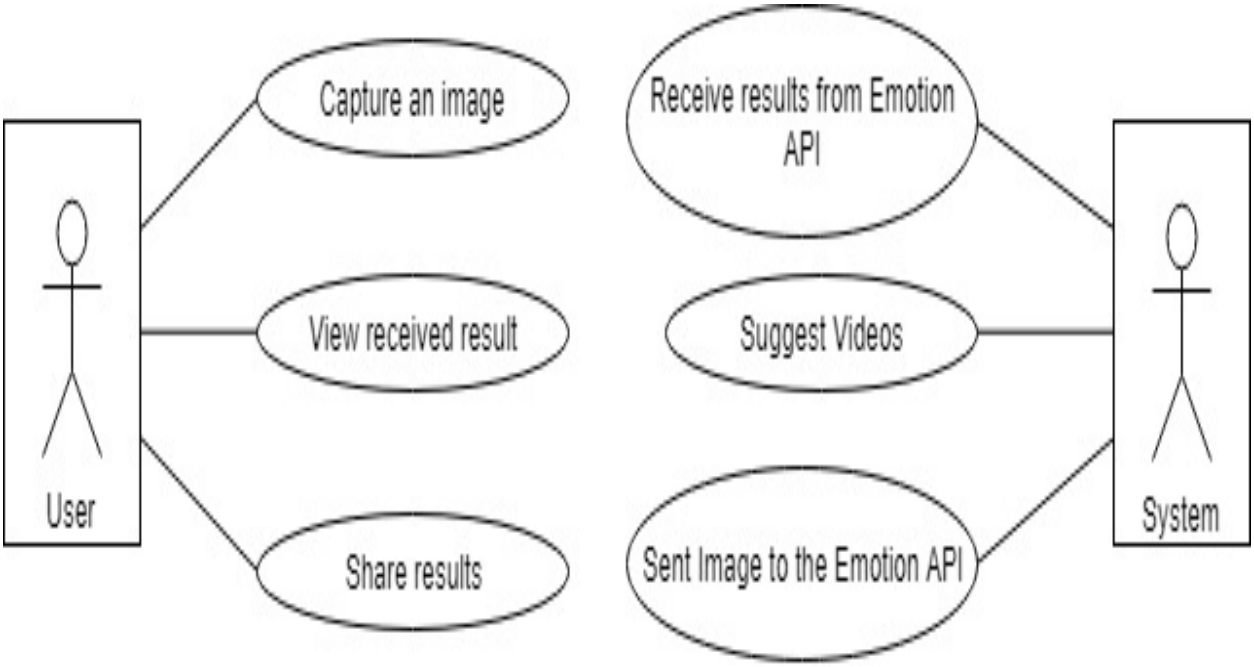


Figure 2- Use Case Diagram

4.1 Use-Case Realizations

The use case realizations will explain how the given use cases actually work in the system. The following use case descriptions will give a clear idea about those functionalities.

- Capture Image

Use case name	Capture Image	
Actor	User	
Description	When the user requests to take an image of his/her face in his/her Android application, the system should get the image from the user.	
Preconditions	The user should have downloaded the Look It Android Application	
Main flow	User	System
	1. User enters the System 2. Request to take an image	1.1 Display front Interface with Camera button 2.1 Take the image 2.2 Display the image for some seconds
Successful end/post condition	User image will be captured	
Fail end/post condition	Notification will be prompted to retry with correct details	
Extensions	N/A	

- View received results

Use case name	View received results	
Actor	User	
Description	When the user requests to view the details of the received results text, the system should display the text of the received result of him/her image in text format.	
Preconditions	The user should have downloaded the Look It Android Application	
Main flow	User	System
	1. User requests to view received results details	1.1 Check for received results text 1.2 Show results details
Successful end/post condition	Result details will be displayed	
Fail end/post condition	Notification will be prompted if there are no any text received	
Extensions	N/A	

- Share results

Use case name	share results	
Actor	User	
Description	When the user requests to share the details of the received results text, the system should share the text of the received result of him/her image in his / her Social media page.	
Preconditions	The user should have the social media account	
Main flow	User	System
	1. User requests to share received results details	1.1 Check for available social media 1.2 Share the results details
Successful end/post condition	Result details will be shared	
Fail end/post condition	Notification will be prompted if there are no any post shared.	
Extensions	N/A	

- Receive results from Emotion API

Use case name	Receive results from Emotion API	
Actor	System	
Description	When the classification is finished at the Emotion API, the results will be received to the Look It application by using the REST API	
Preconditions	User should take a clear image of his/ her face using his Look It application.	
Main flow	User	System
	1. Request to view result	1.1 Get the results using REST API when the classification is over
Successful end/post condition	Results will be received at the REST API.	
Fail end/post condition	Result will not be received, and an error message should be obtained	
Extensions	N/A	

- Suggest Videos

Use case name	Suggest Videos	
Actor	System	
Description	When the classification is finished at the Emotion API, the results will be received to the Look It application by using the REST API. If result shows that user have negative emotions, then system suggest videos	
Preconditions	User have negative emotion when he/she take his/her face image	
Main flow	System	User
	1. System suggest videos according user's negative feeling	1.1 Accept the Suggestion or Reject the suggestion 1.2 If he accepts the suggestion can watch the video
Successful end/post condition	Video will be displayed	
Fail end/post condition	Notification will be prompted if that video is not in YouTube	
Extensions	N/A	

- Sent image to Emotion API

Use case name	Sent image to Emotion API	
Actor	System	
Description	When User take his face image and request to get results, System Sent the image through REST API to Emotion API to get the results	
Preconditions	User should take a clear image of his/ her face using his Look It application.	
Main flow	User	System
	1. Request to view result	1.1 sent the image using REST API to Emotion API to get results
Successful end/post condition	Image will be sent through REST API.	
Fail end/post condition	Image will not be sent, and an error message should be obtained	
Extensions	N/A	

5. Logical View

The logical view section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. And for each significant package, its decomposition into classes and class utilities. Important classes, their relationships, operations, and attributes are also shown.

5.1 Overview

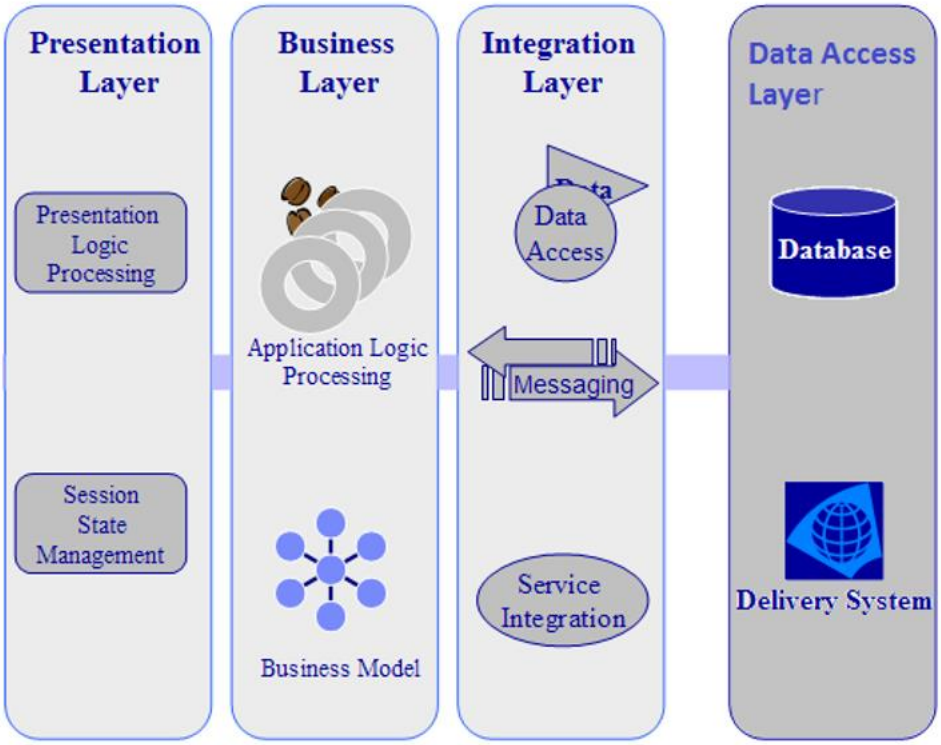


Figure 3- Logical view

5.2 Architecturally Significant Design Packages

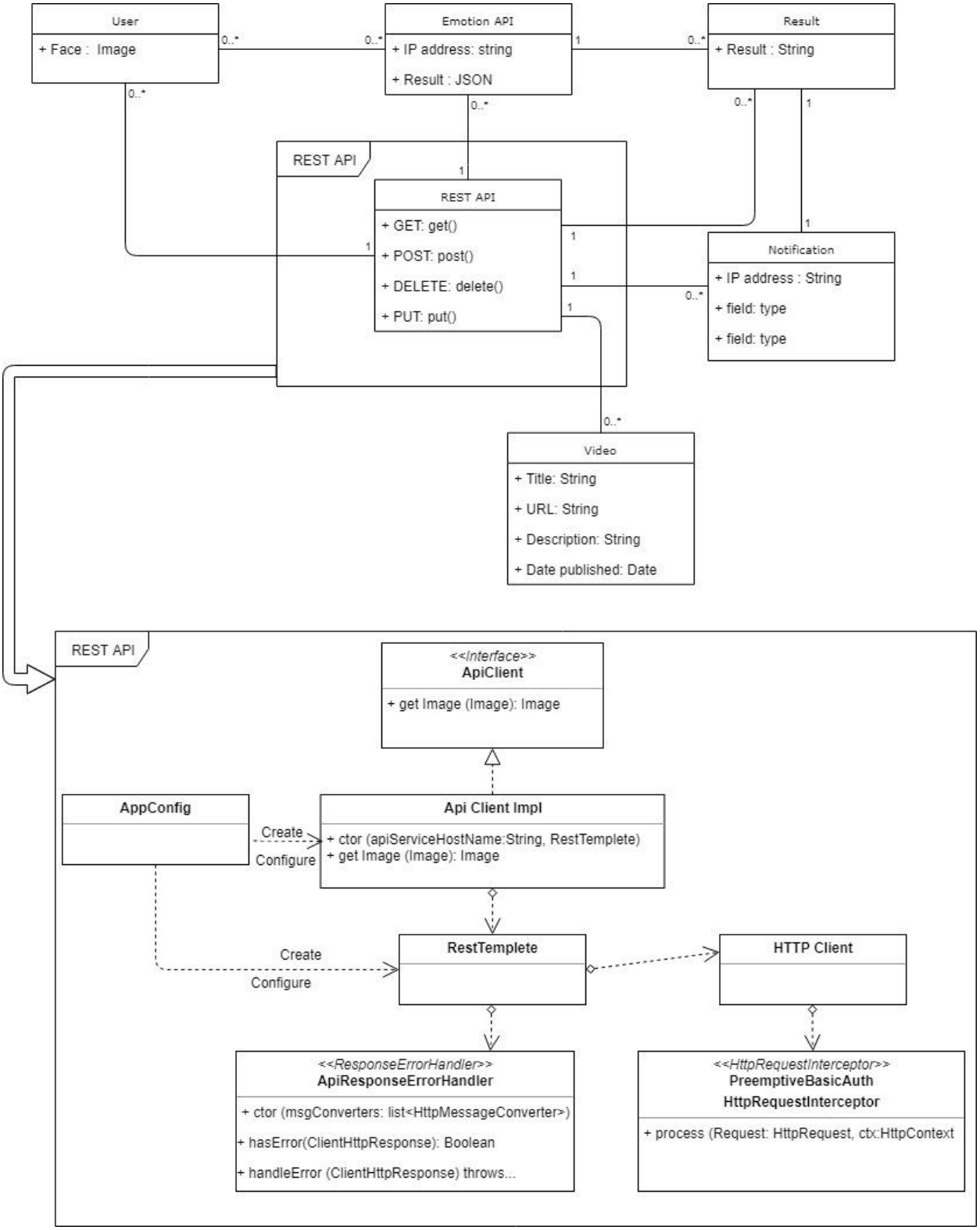


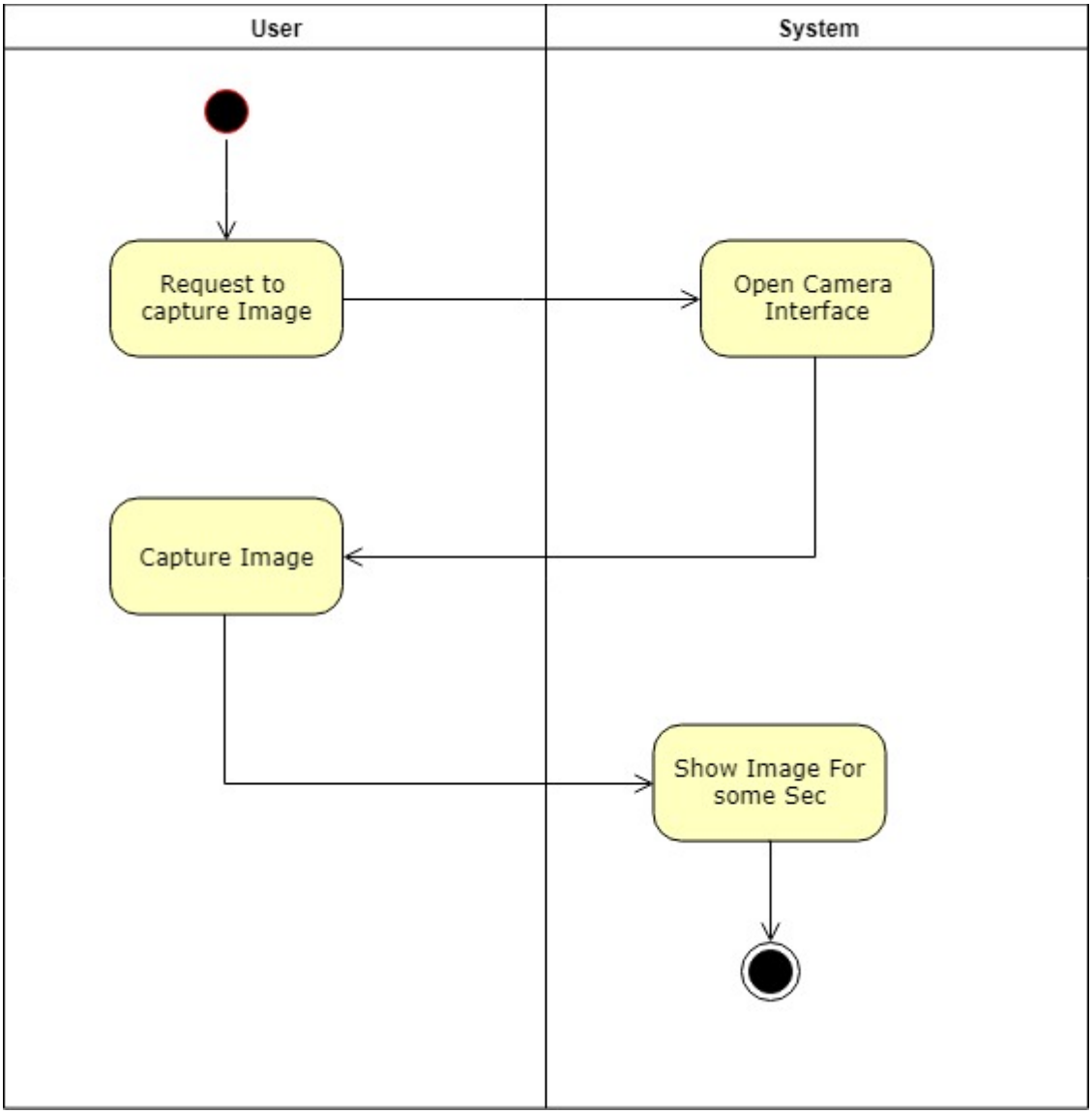
Figure 4- Class Diagram

6. Process View

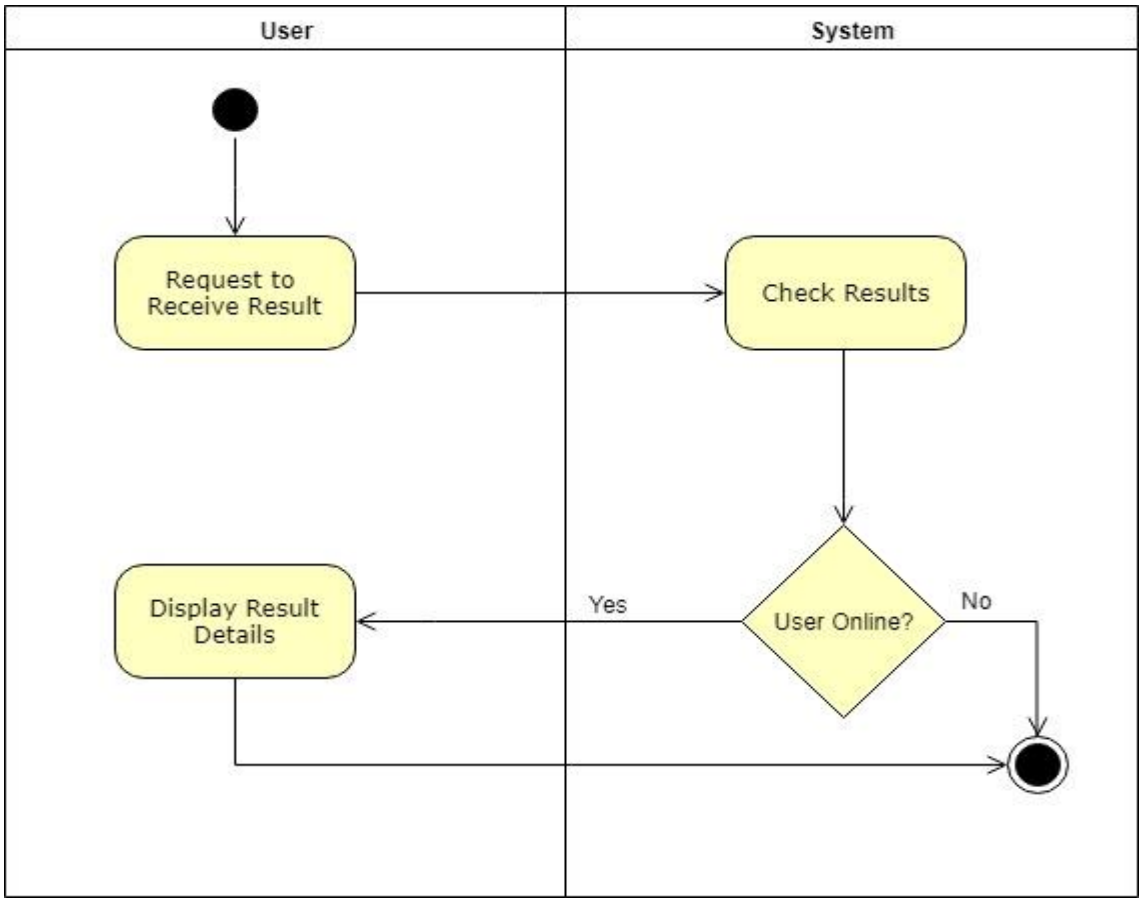
Main processes of the system are explained through the following activity and sequence diagrams.

6.1 Activity Diagrams

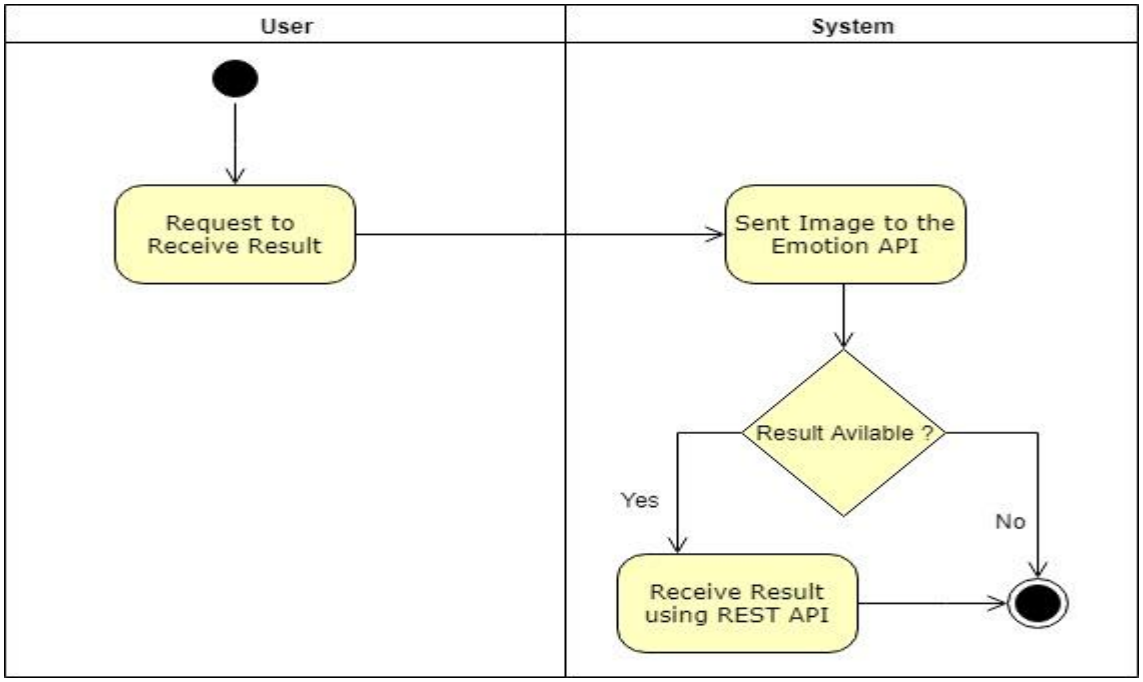
- Capture Image



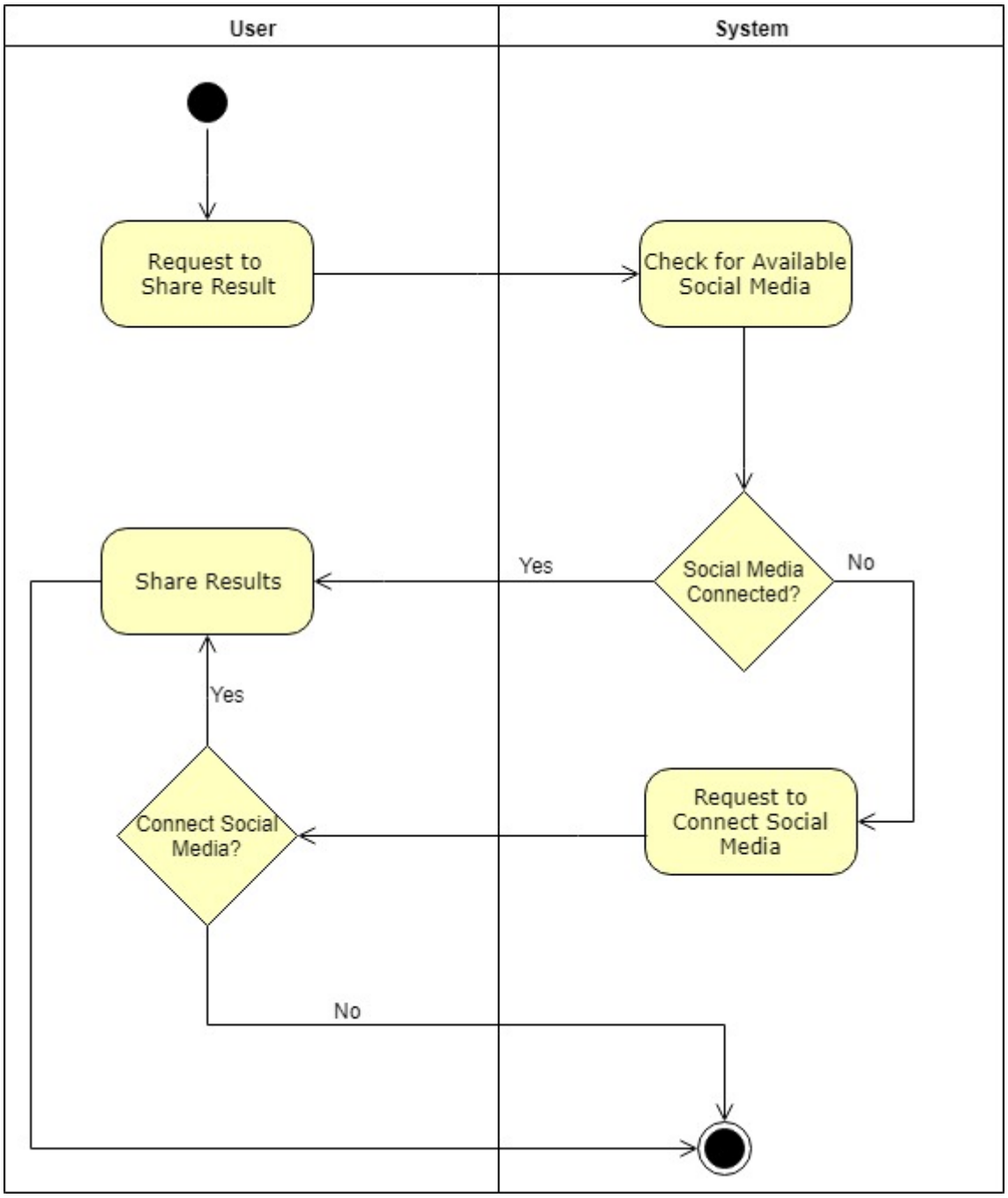
- View received results



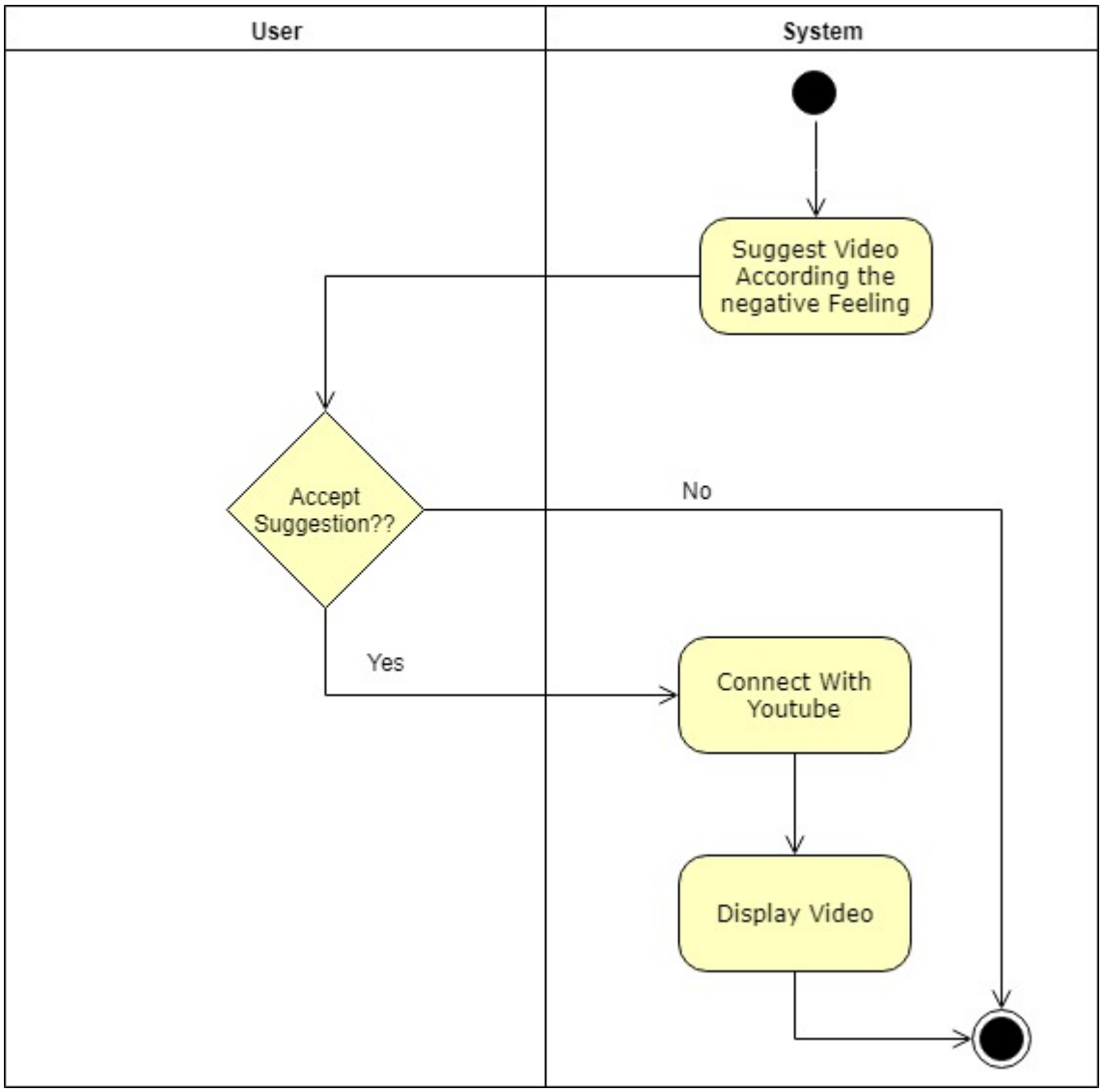
- Receive results from Emotion API



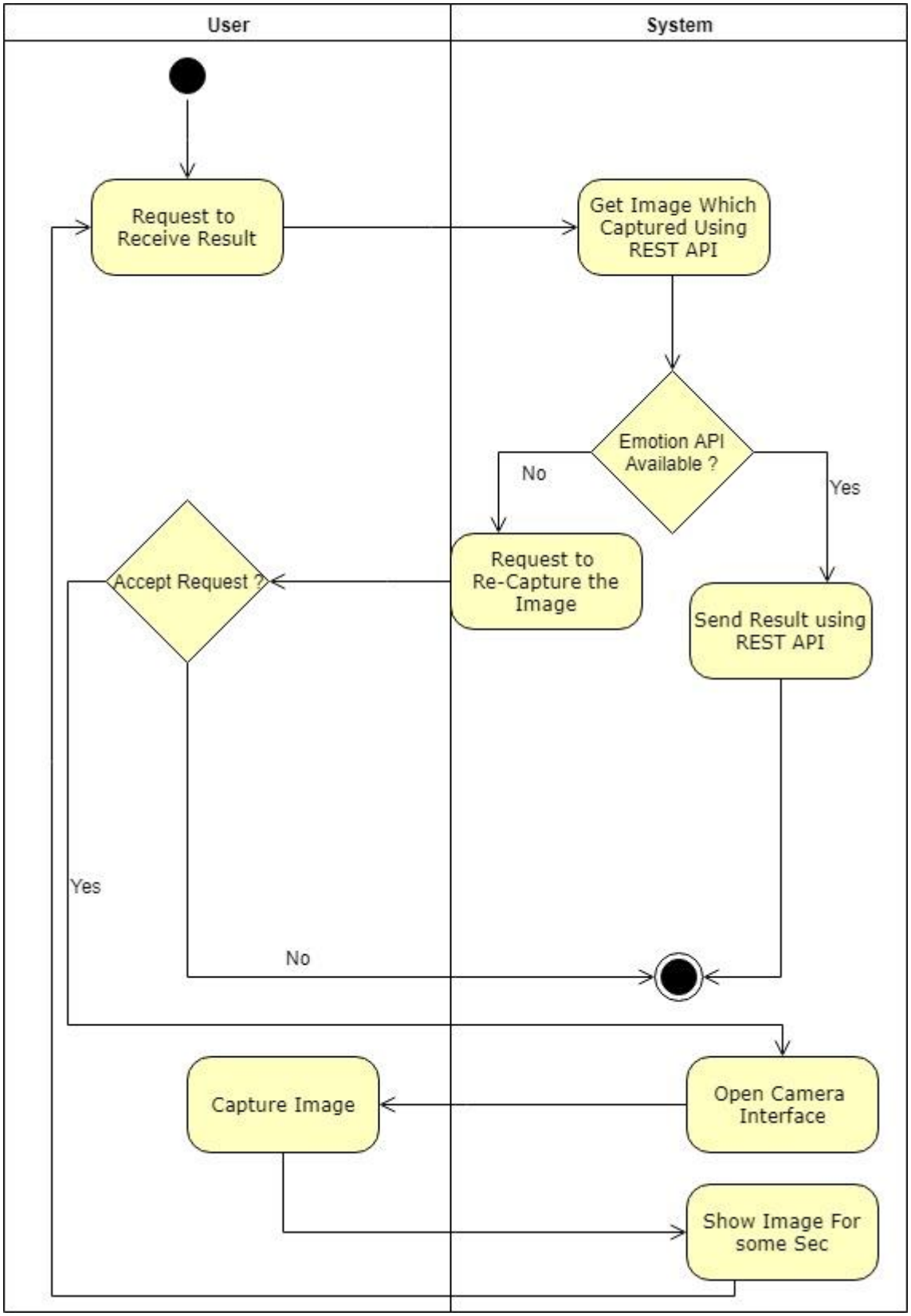
- Share results



- Suggest Videos

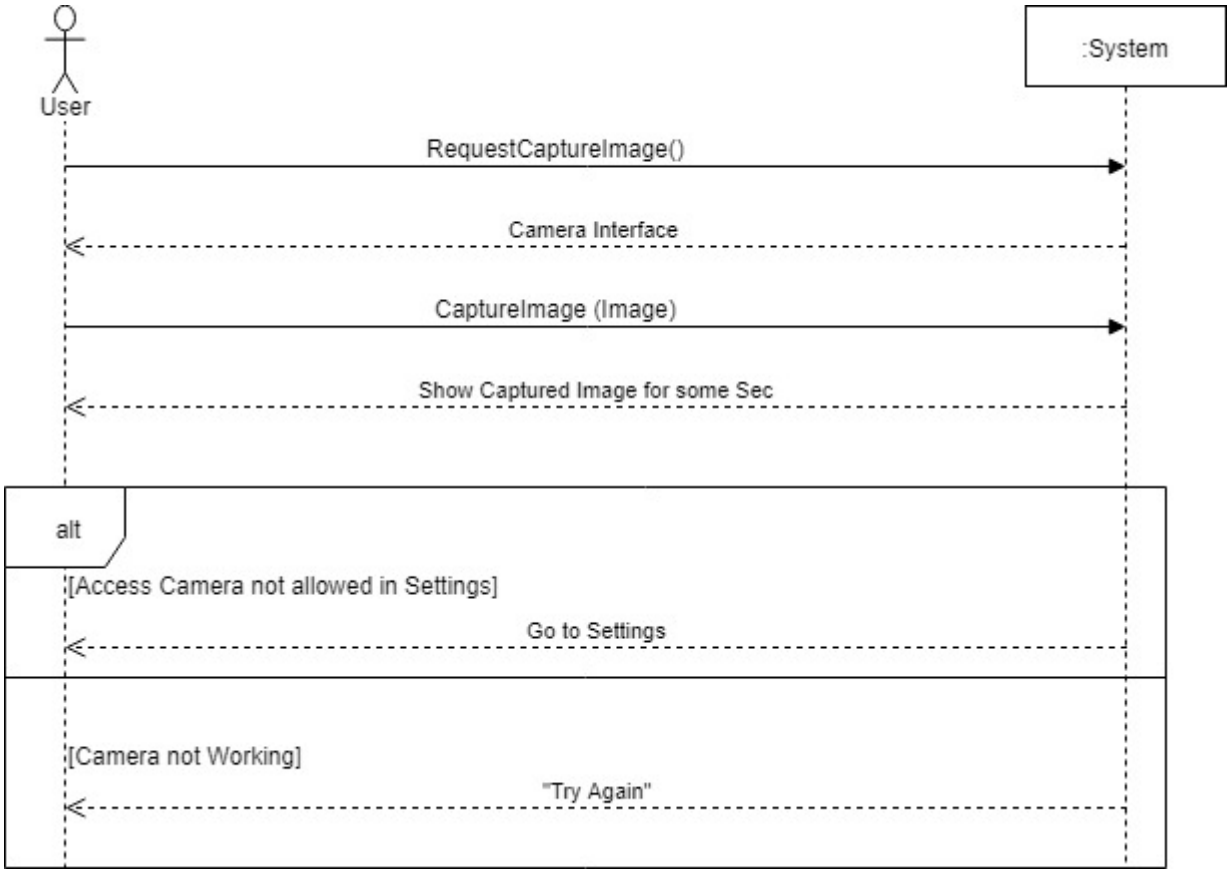


- Sent image to Emotion API

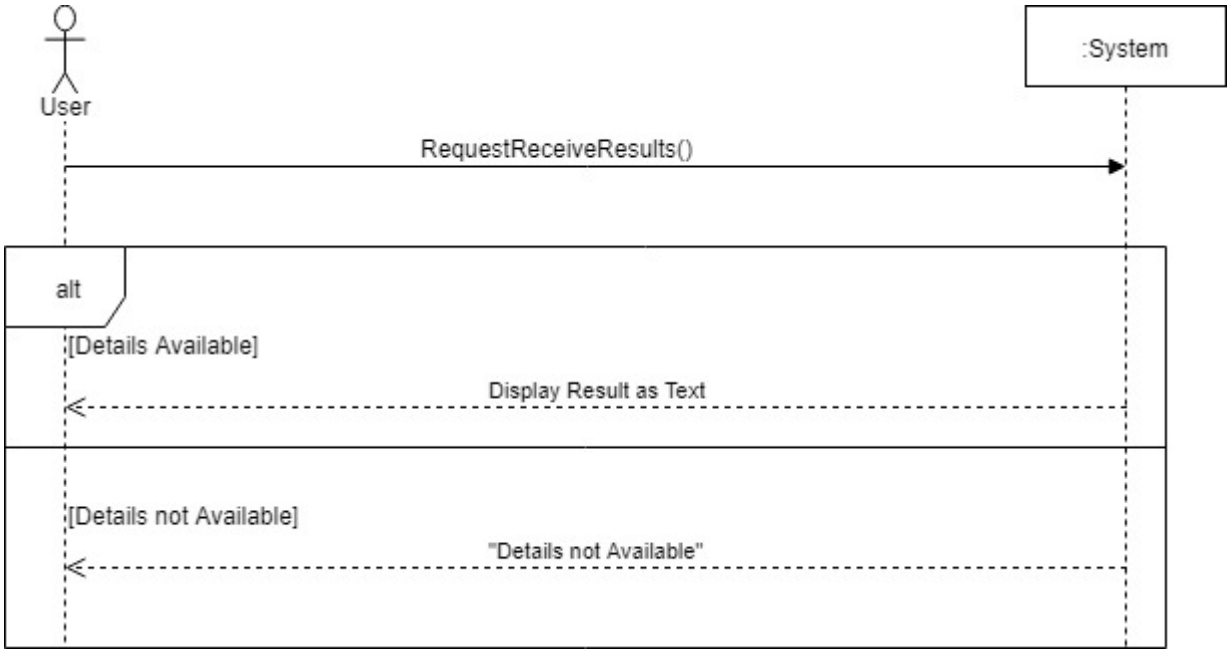


6.2 Sequence Diagrams

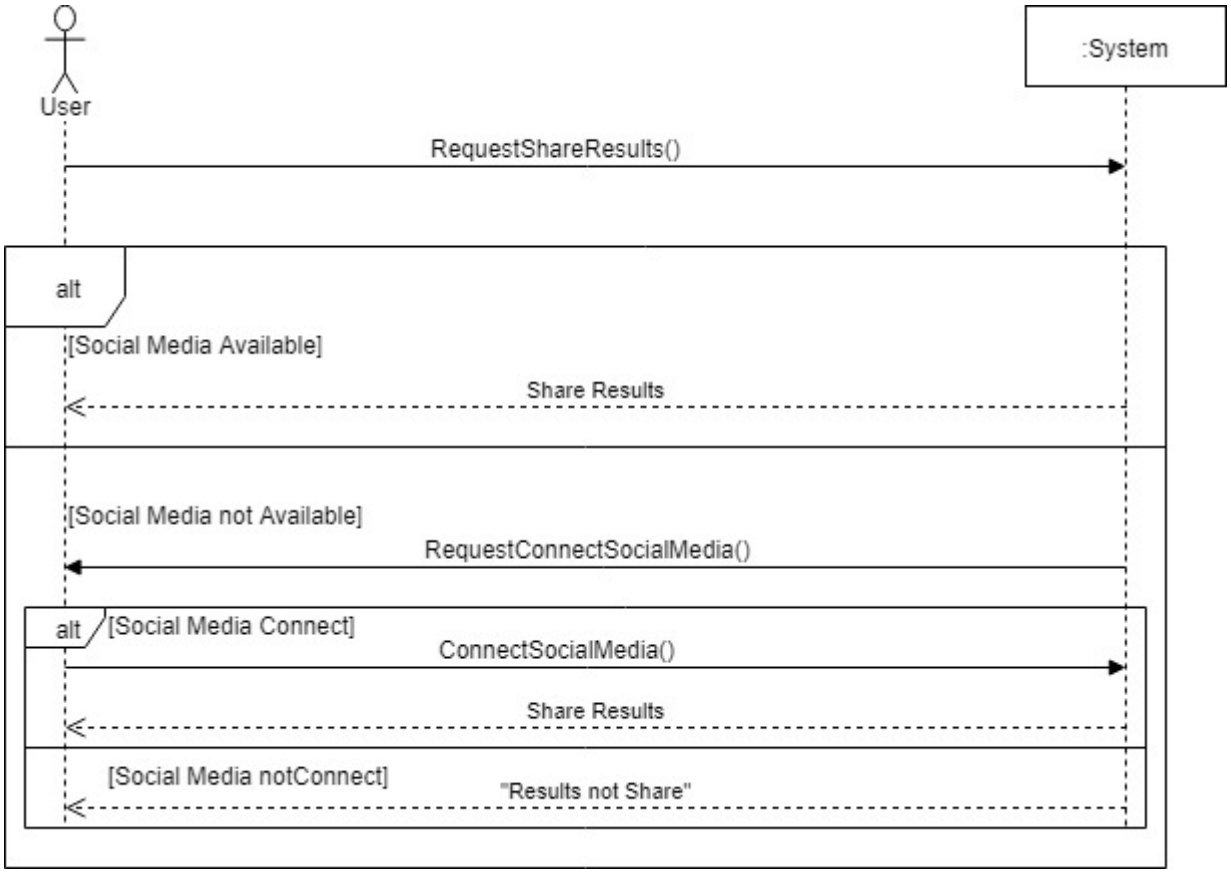
- Capture Image



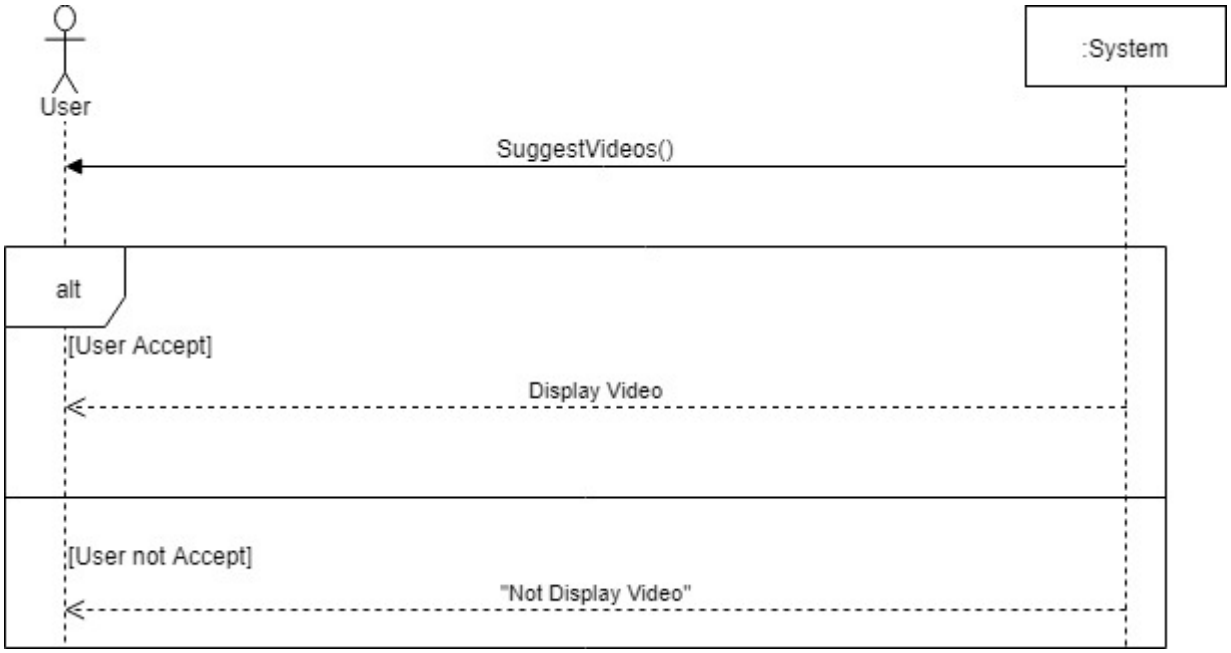
- View received results



- Share results



- Suggest Videos



- Send notification

7. Deployment View

This view provides a clear picture about the high-level architecture of the system. User of the system access it through the internet. Web server does the tasks in order to response the requests. All the user activities of the system will be saved in the system.

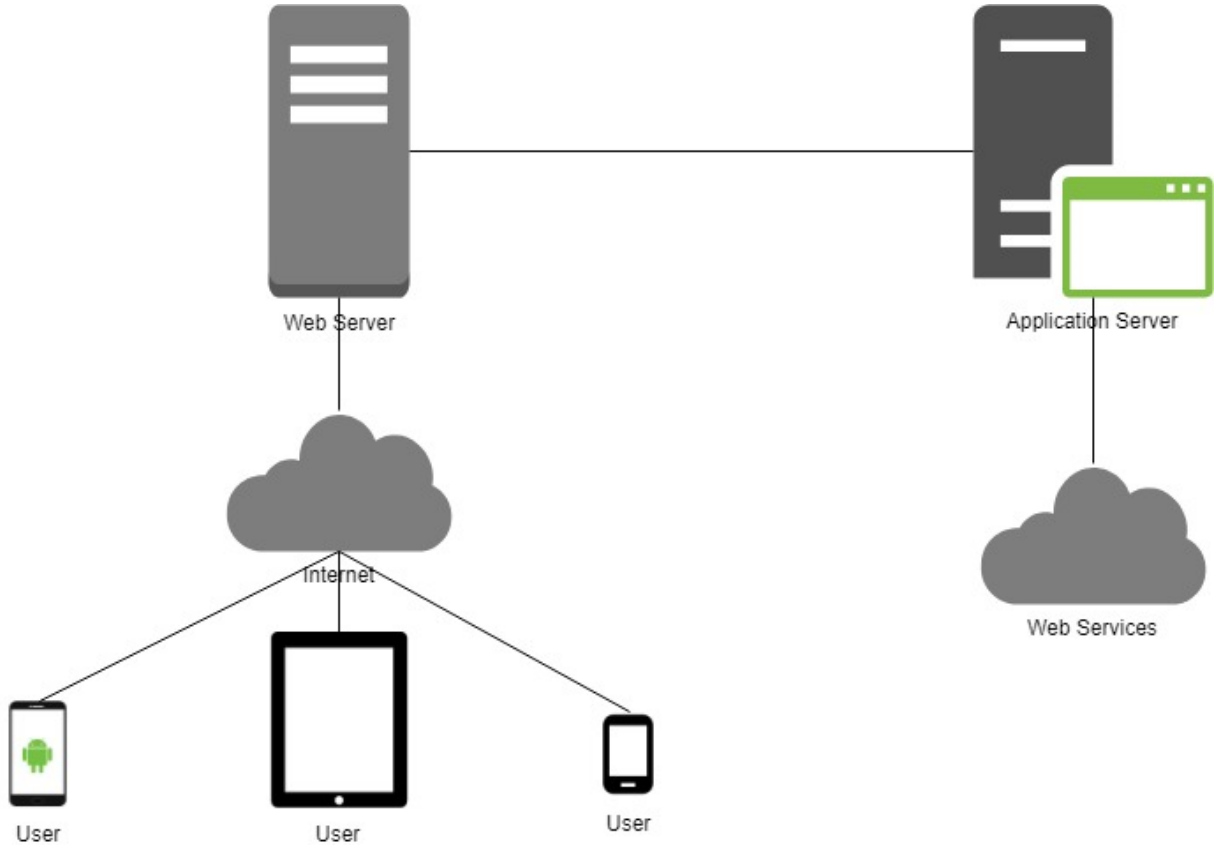


Figure 5- Deployment Diagram

8. Implementation View

8.1 Overview

For the Look It Application, MVC Architecture is used for the Software development. Model – View – Controller Architecture is mostly used in the software industry now.

- Models represent knowledge. A model could be a single object, or it could be some structure of objects.
- A view is a representation of its model. It expresses certain attributes of the model and suppress others. It is acting as a presentation layer.

- Controller is the link between a user and the system. It provides the user with input by arranging for relevant views to present themselves in appropriate places on the screen. It provides means for user output by presenting the user with menus or other means of giving commands and data. The controller receives such user output, translates it into the appropriate messages and pass these messages on to one or more of the views

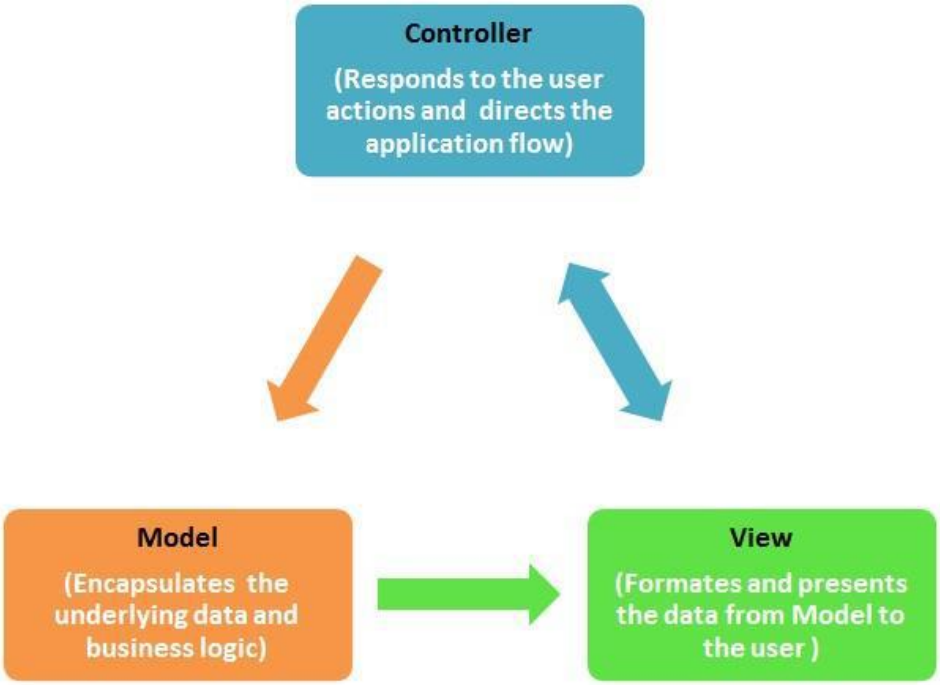


Figure 6- MVC Architecture Diagram

8.2 Layers

- Presentation Layer:
  The presentation layer contains all the front-end user components and the UI that are in end user environment such as Image take interface. It consists of some logic to provide these interfaces as well as to validate the input.

- Web Services Layer:
  The web services layer consists of all the services of the system and provides all the business logic of the system. Moreover, it acts as an intermediate link between presentation layer and the data layer.

- Data Layer:
  The data layer consists of the database. This layer provides persistence for the system and the logic directly related to the manipulation of that data through the means of stored procedures.

9. Size and Performance

Currently the system is developed so as to manage about 100 users at a time. It will be expanded to handle vast number of users in the future.

10. Quality

The system will be having the following qualities.

10.1 Portability

As this is an Android based application, it can be used from anywhere, anytime without any restrictions other than internet connection.

10.2 Security

Proper authentication mechanisms will be followed to make sure the security of the system. Emotion API is Secure.

10.3 Extensibility

This application can be further developed with the user demand and can be supported the demands.