Faculty of Engineering Helwan University

Graduation project 2021-2022

Computer engineering department

Blind Assistant

(See For Me)

Supervised by:

Dr. Hadeer Ahmed

Presented by

Ahmed Hisham Mostafa.

Ali Hassan Ali Mostafa.

Esraa Nageh Omar Ali.

Islam Roshdi Mohamed.

July, 2022

“The only thing worse than being blind is having sight but no vision”

Helen Keller

Contributions

Name	Contributions
Esraa Nageh Omar Ali	1- Clothes detection model 2- Color detection model 3- Banknotes recognition model 4- Object detection model. 5- Handwriting recognition model.
Ali Hassan Ali Mostafa	1- Face recognition model 2- Scene description model 3- Handwriting recognition model. 4- Text recognition model. 5- Object detection model
Ahmed Hisham Mostafa	1- Video call app 2- Hardware feature (tracking) 3- Testing Phase 4- Application integration 5- Camera app 6- Features Blind user

	interface. 7- UI/UX for application. 8- Sequence diagrams 9- Use-case diagrams
Islam Rushdi Mohamed Mohamed	1. Banknotes rec. server 2. Clothes detection server 3. Color recognition server 4. Face recognition server 5. Object detection server 6. OCR server 7. Scene description server 8. Video call server 9. Client-side socket for each feature. 10.System integration (Connection the model to the server to the client side) 11.sequence diagrams 12.Entity-relationship diagram 13.realtime database

Abstract

This book will focus on our team's Graduation project and the main objective of this project is to make the life of blind people easier using a collaboration of Software, AI, and Hardware.

After doing some research, we found that most engineers tried to make a piece of technology to help blinds. Is by creating a mobile application that helps them with everyday activities such as calling someone or choosing to use some kind of stick or glasses to avoid accidents or recognize the objects for the blind. As a consequence, we decided to combine all of those with our - (See For Me) mobile APP with the Ultrasonic sensor to introduce a complete application that will help blinds function in their real-life lives in an easier way without social awkwardness, accidents, and with more fun and exploration.

Acknowledgments:

We would like to express our sincere gratitude for supporting us throughout our Graduate project. We wish to express our sincere gratitude to our supervisor **Dr. Hadeer Ahmed** for her enthusiasm, patience, insightful comments, helpful information, practical advice, and unceasing ideas that have helped us tremendously at all times in our project. Her immense knowledge, profound experience, and professional expertise have enabled us to complete this project successfully. Without her support and guidance, this project would not have been possible.

Ahmed, Ali, Islam, and Esraa

Table of contents:

Contents

Table of figures:

Chapter 1

Chapter 1: Introduction:

Some researchers imply that the **Struggles of the visually-impaired:**

Over **39** million are blind with **246** million low vision and nearly **90%** of them live with low Income.
They faced many struggles like limited access to activities and information and social stigma. As a consequence, that was the motivation for us to work on the Blind Assistant (See For Me) application using the combination of the AI software and hardware technologies to serve their needs with the help of a machine (mobile-phone).

1.1. Motivation:

We need to help visually impaired people interact more independently with the external world.

1.2 Project Idea:

This application is easy to use, and it can be used by users of different ages. There are two main parts to our project: the software part and the hardware part. The hardware part will contain an Ultrasonic sensor. While the software part will be Open-CV as the framework, that uses Python as the programming language. We also use a deep learning technique. This technique is a type of machine learning method that uses neural network architectures to extract features from an image in order to recognize the object that exists in the captured image. We have four modes in our project, which are App**.** assistant mode, Volunteer mode, Walk mode, and Navigation mode.

First mode: App. assistant mode it has several features:

- Object detection feature to detect the objects in the image that are captured and helps him to detect objects if he goes out or the objects are moved from the place the blind person saves them in.
- Text recognition feature detects and extracts text from any image.
- Color classification feature that can recognize the color in the image that is captured to help the blind person in choosing the clothes or recognize the nature around him.
- Face recognition feature that makes the blind person recognize his family and friends. For example, if he goes out to meet one of his family or friends he can search for him by capturing an image and search for his friend in it, but how would the feature know it? Friends and family members of the blind person will save an image of each one of them and save each one in the application by his name.
- Clothes detection feature helps the blind person in shopping so that he can search for any type of clothes in the image captured and also help him in choosing his clothes before going out.
- Scene recognition feature helps the blind person to recognize the place that he is in by taking an image and knowing the place for example bathroom, bedroom, etc.
- Banknotes recognition feature, when the blind person buys anything. This feature prevents him from scammers and thieves, this feature can recognize the banknotes in the image captured by the blind person.

Second mode: Volunteer mode it has video call feature:

In video call feature, the blind person can ask for help from a volunteer to help him in case the previous features cannot help the blind person with something. So he has the solution for his issue.

Third mode: Walk mode which helps the blind person to walk safely. As the sensor device connecting to the app that senses any object that causes a danger during walking will alert the blind.

Fourth mode: Navigation mode which helps the blind person to go to any location by using the Google maps application.

Chapter 2

Chapter 2: Related Work:

In previous work many devices were made for the blind person to help him in his daily life, from these devices are the Smart glasses, Pen friend, and Smart stick.

2.1 Smart glasses:

Designed for the blind and visually impaired as presented in figure 1. It uses artificial intelligence (AI) to extract information from images and verbally tell its wearer what they are looking at. [1]

The AI-powered glasses enable blind and low-vision users to read documents at work, and use public transportation.

Disadvantages of smart glasses:

- Limited by performance and accuracy of hardware.
- Cost is so high for blind people that they probably don't work so most of them can't buy a device with this cost.

Figure 1: Smart Glasses Device

2.2 Pen friend:

The pen friend is a voice labeling system that uses a simple device to create recordings associated with sticker labels as shown in figure 2. The user taps a sticker with the Pen Friend, makes a voice recording, then the Pen Friend saves the voice recording for that sticker. It is designed to label food items and household objects. [2]

Disadvantages of pen friend:

- Has just one feature of text recognition.
- Cost is high.

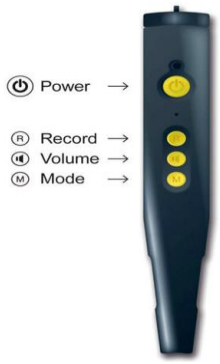


Figure 2: Pen Friend Device

2.3 Smart Stick:

Smart stick is basically an embedded system integrating the following: pair of the ultrasonic sensor to detect obstacles in front of the blind from ground level height to head level height in the range of 400 cm a head, and an infrared sensor to detect upward and downward stairs. [3]

As presented in figure 3.

Disadvantages of smart stick:

- It is made for one feature that is tracking.
- The cost is so high for blind people.

Figure 3: Smart Stick Device

From the previous related work, we notice some notes:

- The cost is so high.
- All of these works are made for just one feature.
- Why not use smartphones?

After these notes, we searched for applications and tried to solve the disadvantages of previous work and found the following applications:

2.4Intelligent eye:

It is an application that provides assistance to visually impaired people as presented in figure 3by providing a set of features: light detection, color detection, object recognition, and banknote recognition. [4]

Disadvantages is that the bottoms are very close to each other which would be a problem for the blind person.

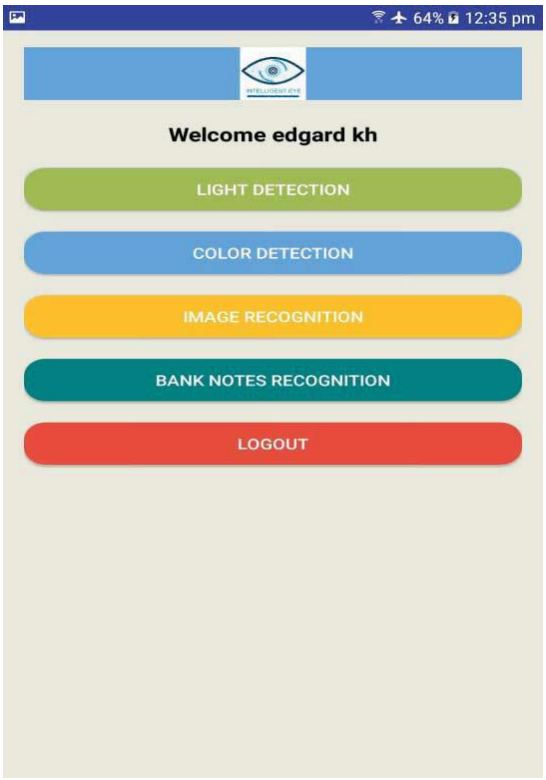


Figure 4: Intelligent Eye Application Interface

2.5Be my eyes application:

Be My Eyes is another application that connects the blind person with a global community of volunteers and company representatives who are ready at a moment's notice to help the blind see. [5]

Disadvantages of be my eyes application:

- It is made for just one feature.
- The blind person will feel that he needs help that does not exist in our solutions for the blind.

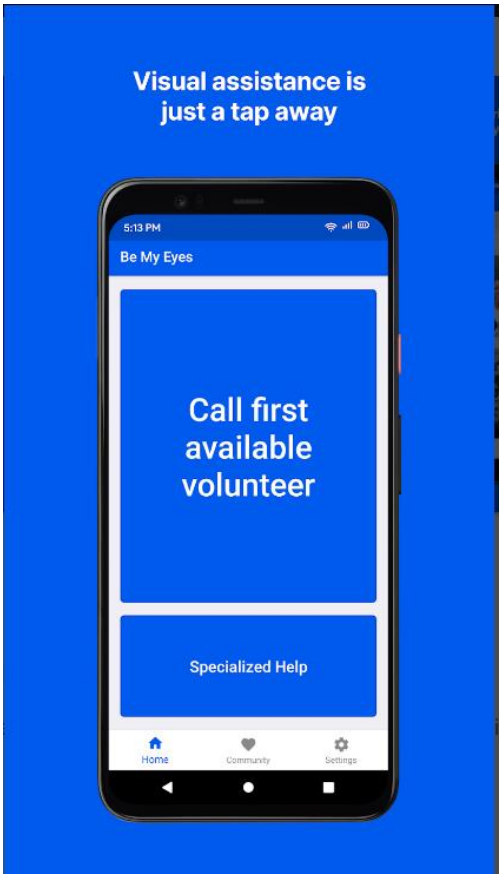


Figure 5: By My Eyes Application

Chapter 3

Chapter 3: Analysis Requirements.

3.1 System Requirement

3.1.1 Functional Requirements:

Video call	REQ.1	Blinds can look for volunteers who can help them in something that does not exist in the application.
Face recognition	REQ.2	Helping him find his friends using only the mobile phone camera. Recognize and manipulate faces.
Object detection	REQ.3	Detect objects to blind in any captured photo.
Clothes detection	REQ.4	Detects clothes to blind to help him in shopping.
Text recognition	REQ.5	Detect the text words and lines and help the blind to read texts.
banknotes recognition	REQ.6	Detect the value of the banknotes.
color recognition	REQ.7	Helping blind recognize the color of different items.

3.1.2 Nonfunctional requirements:

REQ-9 (PERFORMANCE)	● Provide faster software than other applications. ● Fast detection. ● Generalized application.
REQ-9 (SAFETY)	● Protect blind from scamming. ● Protect blind from accidents.
REQ-10 (SECURITY)	● Account ID and Password (PIN) Protection. ● Auto Timeout Screen Blanking. ● Sign-off Button Failed Log-on Attempts. ● Encryption.

3.2 Functional Requirement Specification

3.2.1 Stakeholders:

Blind: the blind person who interacts directly with the application that the application made for him for helping him in his life.

Volunteer: Use the application that the blind can need him for help and he volute to help the blind.

Server: server which handles the operation of the application.

3.2.2 Actors and goals:

Blind: the blind person who interacts directly with the application that the application made for him for helping him in his life.

3.2.3 Use-Case Diagram:

Figures 6 provides the details description of the overall use case diagram, and figure 7 is the details description of video call use case diagram.

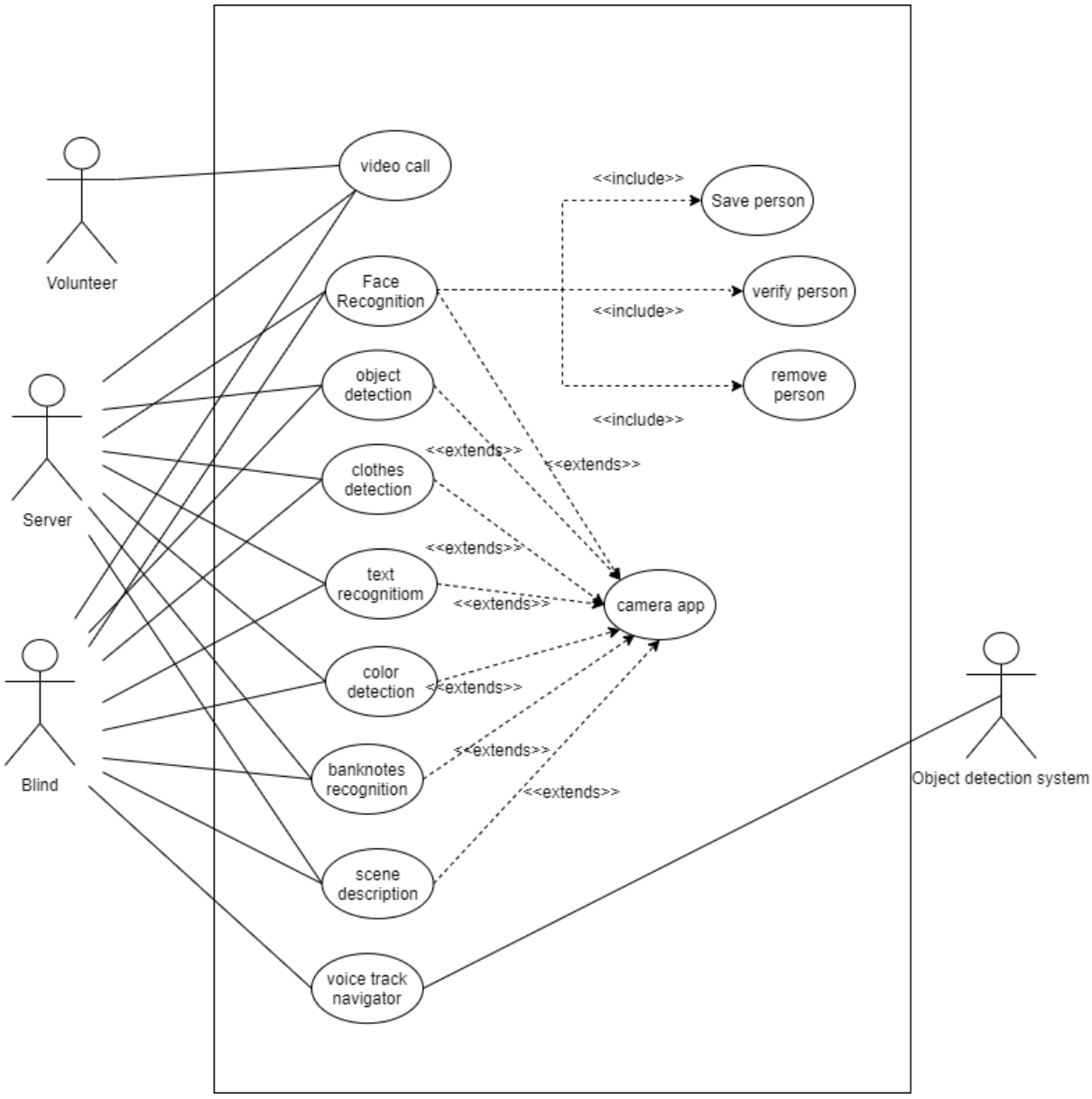


Figure 6: Application Use Case Diagram

Figure 7: In Details Video Call Use Case Diagram

Chapter 4

Chapter 4: Design

4.1 Block diagram:

Description of proposed application

The main idea of our application is that the Blind person will communicate with the app by his voice as presented in figure 8.

Figure 8: Voice Assistant Interface

As presented in figure 8 the blind can use his voice and ask for application help and take an image the image sent to server and enter as an input to trained model, the model output the predication and then received to the blind as speech.

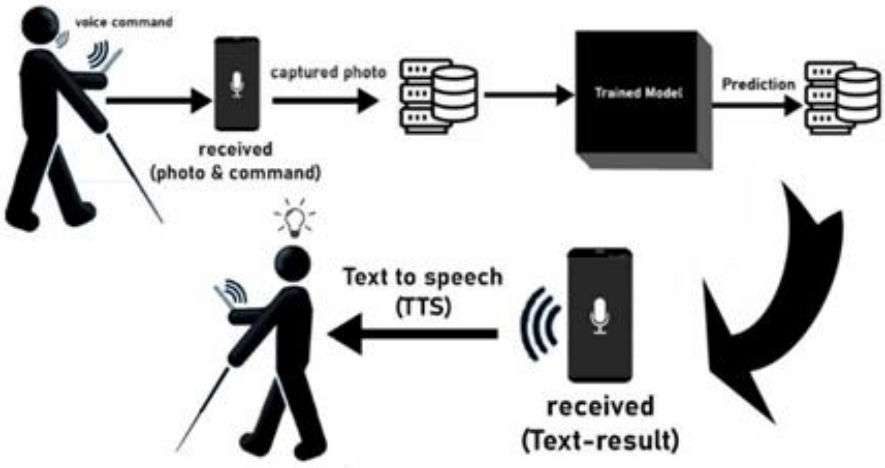


Figure 9: Voice Assistant life cycle diagram

Our proposed application provides the blind with many features which are Object detection, Face recognition, Scene recognition, Text recognition, Clothes detection, Color detection, and Banknotes recognition.

Advantages of the proposed application:

Our proposed application provided the blind with many facilities that make it a good choice for the blind to use:

- Easy to use for the blind person.
- Many features in one application.
- Cheap, efficient, no help required.
- Covering many cases that face the blind person.
- Indoor and outdoor helper.
- Remote monitoring so it gives the partners of blind people the ability to be away for some time without worrying about any emergencies.

4.2 Sequence Diagram:

-As presented in Figures 10,11 AI features sequence diagrams

Here we describe the sequence of the user using any feature including an AI model, as well as the sequence of calling volunteer with video call mode.

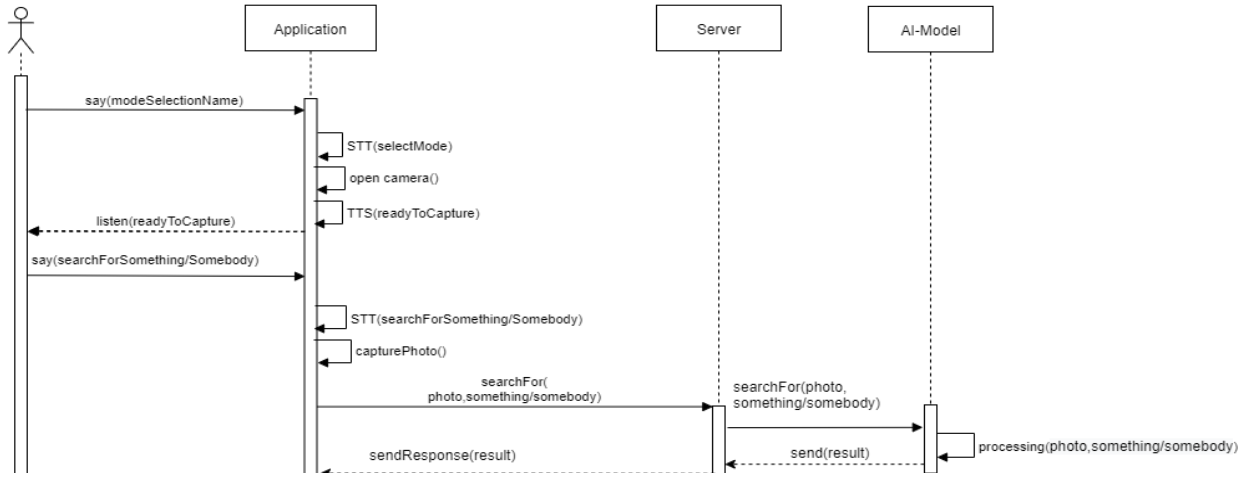


Figure 10: Application AI Features Sequence Diagram

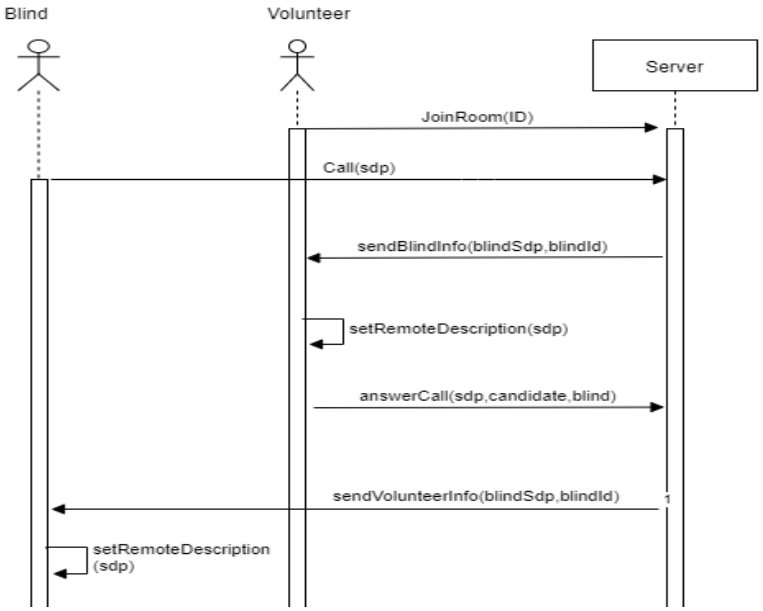


Figure 11: Video Call Sequence Diagram

4.3 Entity Relationship Diagram (ERD):

-As shown in Figure 12 the ER diagram of the application.

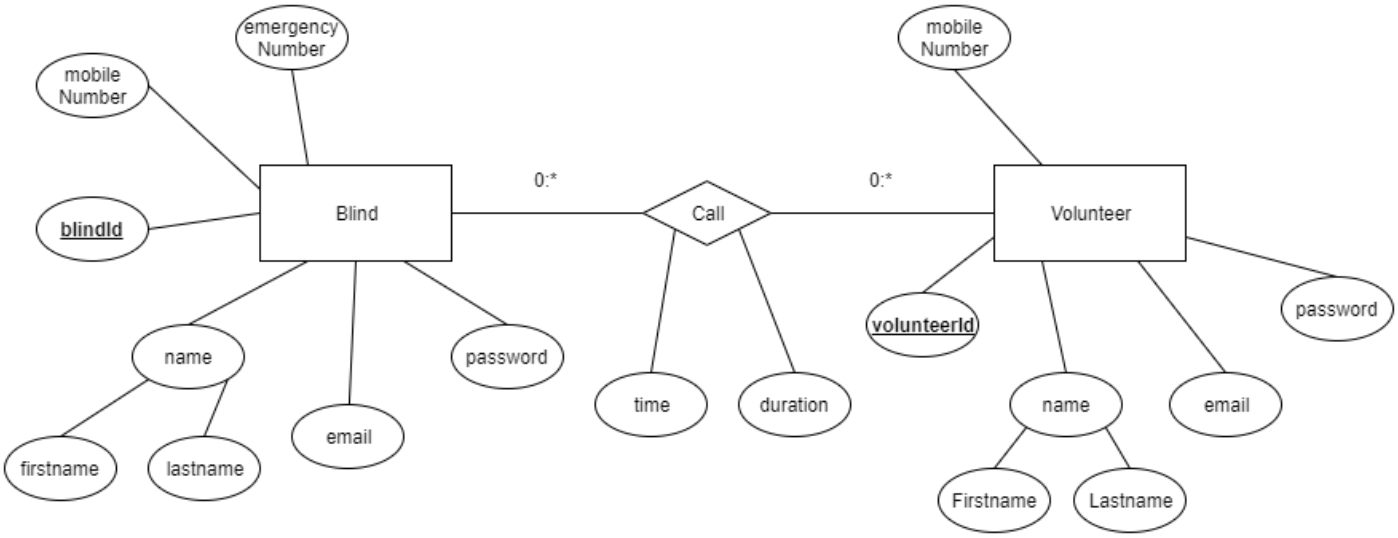


Figure 12: Application Entity Relationship Diagram

Chapter 5

Chapter 5: Testing and Validation:

Validation and testing is the process of ensuring if the tested and developed software satisfies the client /user needs. The business requirement logic or scenarios must be tested in detail. All the critical functionalities of an application must be tested here.

Verification	Validation
The process just checks the design, code, and program.	It should evaluate the entire product including the code.
Reviews, walkthroughs, inspections, and desk- checking involved.	Functional and nonfunctional methods of testing are involved. In depth check of the product is done.
It checks the software with specification.	It checks if the software meets the user needs.

So in order to perform testing we must achieve two concepts

Black Box Testing	White Box Testing
Black box testing is the Software testing method which is used to test the software without knowing the internal structure of code or program.	White box testing is the software testing method in which internal structure is being known to tester who is going to test the software.
This type of testing is carried out by testers.	Generally, this type of testing is carried out by software developers.
Implementation Knowledge is not required to carry out Black Box Testing.	Implementation Knowledge is required to carry out White Box Testing.
Programming Knowledge is not required to carry out Black Box Testing.	Programming Knowledge is required to carry out White Box Testing.
Testing is applicable on higher levels	Testing is applicable on lower level of

of testing like System Testing, Acceptance testing.	testing like Unit Testing, Integration testing.
Black box testing means functional test or external testing. In Black Box testing is primarily concentrate on the functionality of the system under test.	White box testing means structural test or interior testing. In White Box testing is primarily concentrate on the testing of program code of the system under test like code structure, branches, conditions, loops etc.
The main aim of this testing to check on what functionality is performing by the system under test.	The main aim of White Box testing to check on how System is performing.
Black Box testing can be started based on Requirement Specifications documents.	White Box testing can be started based on Detail Design documents.
The Functional testing, Behavior testing, Close box testing is carried out under Black Box testing, so there is no required of the programming knowledge.	The Structural testing, Logic testing, Path testing, Loop testing, Code coverage testing, Open box testing is carried out under White Box testing, so there is compulsory to know about programming knowledge.

5.1 Black Testing (Using Equivalence Partitioning)

Recognition and Detection Testing

Detection & Recognition Modes Test Cases	Number Of Cameras (cameras)	Camera Controller (controller)	Socket Connection (SC)	Text To Speech Variable (TTS)	Object Name	Image	Speech To Text Variable (STT)	Output
Test Case 1	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = Initialized & has valid data	Output (speech: Laptop)
Test Case 2	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = Initialized & has Invalid data	Error
Test Case 3	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = Initialized & has null value	Error
Test Case 4	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = not initialized	Error
Test Case 5	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has invalid value (laptop scene.jpg)	/	Error
Test Case 6	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has null value	/	Error
Test Case 7	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has Invalid value (2)	/	/	Error
Test Case 8	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = Initialized & has valid data	Output (enter object name first)
Test Case 9	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = Initialized & has Invalid data	Error
Test Case 10	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = not initialized	Error
Test Case 11	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = Initialized & has null value	Error
Test Case 12	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized with no data	/	/	/	Error
Test Case 13	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = not initialized	/	/	/	Error

Test Case 14	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = invalid initialization	/	/	/	Error
Test Case 15	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (send data only)	/	/	/	/	Error
Test Case 16	cameras = 2 (0,1)	controller = connected to camera (0)	SC = connected (Receive data only)	/	/	/	/	Error
Test Case 17	cameras = 2 (0,1)	controller = connected to camera (0)	SC = invalid Connection	/	/	/	/	Error
Test Case 18	cameras = 2 (0,1)	controller = connected to camera (0)	SC = disconnected	/	/	/	/	Error
Test Case 19	cameras = 2 (0,1)	controller = connected to camera (1)	/	/	/	/	/	Error
Test Case 20	cameras = 2 (0,1)	Controller = disconnected	/	/	/	/	/	Error
Test Case 21	cameras = 2 (0,1)	Controller = invalid Connection	/	/	/	/	/	Error
Test Case 22	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = Initialized & has valid data	Output (speech: Laptop)
Test Case 23	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = Initialized & has Invalid data	Error
Test Case 24	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = Initialized & has null value	Error
Test Case 25	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has valid value (laptop scene. base64)	SST = not initialized	Error
Test Case 26	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has invalid value (laptop scene.jpg)	/	Error
Test Case 27	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has valid value (Laptop)	has null value	/	Error
Test Case 28	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has Invalid value (2)	/	/	Error
Test Case 29	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = Initialized & has valid data	Output (enter object name first)

Test Case 30	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = Initialized & has Invalid data	Error
Test Case 31	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = not initialized	Error
Test Case 32	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized & has data	has null value	/	SST = Initialized & has null value	Error
Test Case 33	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = Initialized with no data	/	/	/	Error
Test Case 34	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = not initialized	/	/	/	Error
Test Case 35	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data / receive data)	TTS = invalid initialization	/	/	/	Error
Test Case 36	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (send data only)	/	/	/	/	Error
Test Case 37	cameras = 1 (0)	controller = connected to camera (0)	SC = connected (Receive data only)	/	/	/	/	Error
Test Case 38	cameras = 1 (0)	controller = connected to camera (0)	SC = invalid Connection	/	/	/	/	Error
Test Case 39	cameras = 1 (0)	controller = connected to camera (0)	SC = disconnected	/	/	/	/	Error
Test Case 40	cameras = 1 (0)	controller = connected to camera (1)	/	/	/	/	/	Error
Test Case 41	cameras = 1 (0)	Controller = disconnected	/	/	/	/	/	Error
Test Case 42	cameras = 1 (0)	Controller = invalid Connection	/	/	/	/	/	Error
Test Case 43	Cameras = 1(1)	/	/	/	/	/	/	Error
Test Case 44	Cameras = 0 (null)	/	/	/	/	/	/	Error
Test Case 45	Cameras = invalid value	/	/	/	/	/	/	Error

Video Call Testing

Video Call (In Call) Test Cases	Socket Connection = SC	Local Video Stream Renderer = LR	Remote Video Stream Renderer =RR	Mic	Headset	Switch Camera	Video	Decline	Output
Test Case 1	SC = connected (Peer to peer)	initialized & has value	initialized & has value	enabled/disabled	enabled	Camera [0]/ Camera [1]	enabled/ disabled	enabled/disabled	Output
Test Case 2	/	/	/	/	/	/	/	null	Error
Test Case 3	/	/	/	/	/	/	null	/	Error
Test Case 4	/	/	/	/	/	null	/	/	Error
Test Case 5	/	/	/	/	null	/	/	/	Error
Test Case 6	/	/	/	null	/	/	/	/	Error
Test Case 7	/	/	null	/	/	/	/	/	Error
Test Case 8	/	Null	/	/	/	/	/	/	Error
Test Case 9	Null	/	/	/	/	/	/	/	Error
Test Case 10	SC = Connected (Peer to peer)	initialized & has value	initialized & has no value	/	/	/	/	/	Error
Test Case 11	SC = Connected (Peer to peer)	initialized & has value	initialized & has invalid value	/	/	/	/	/	Error
Test Case 12	SC = Connected (Peer to peer)	initialized & no value	/	/	/	/	/	/	Error
Test Case 13	SC = Connected (Peer to peer)	initialized & has invalid value	/	/	/	/	/	/	Error
Test Case 14	SC = Invalid Connection	/	/	/	/	/	/	/	Error
Test Case 15	SC = Disconnected	/	/	/	/	/	/	/	Error

Test Case 16	SC = connected (Peer to peer)	initialized & has value	initialized & has value	enabled	disabled	Camera [0]/ Camera [1]	enabled/ disabled	enabled/disabled	Error

Hardware Smart Shoes (application part) Testing

Smart Shoes Feature Test Cases	Bluetooth Connection = BC	Device Connection = DC	Device Name	TTS	Current State	Alarm Sound	Output
Test Case 1	BC = connected	DC = connected	name = "HC-06"	initialized & has data	GOOD	Disabled	Output
Test Case 2	BC = connected	DC = connected	name = "HC-06"	initialized & has data	GOOD	Enabled	Error
Test Case 3	BC = connected	DC = connected	name = "HC-06"	initialized & has data	DANGER	Disabled	Error
Test Case 4	BC = connected	DC = connected	name = "HC-06"	initialized & has data	DANGER	Enabled	Output
Test Case 5	BC = connected	DC = connected	name = "HC-06"	initialized & has invalid data	/	/	Error
Test Case 6	BC = connected	DC = connected	name = "HC-06"	initialized & with null value	/	/	Error
Test Case 7	BC = connected	DC = connected	name = "HC-06"	invalid initialization	/	/	Error
Test Case 8	BC = connected	DC = connected	name = "HC-06"	not initialized	/	/	Error
Test Case 9	BC = connected	DC = connected	name = other name than "HC-06"	/	/	/	Error
Test Case 10	BC = connected	DC = connected	name = null	/	/	/	Error
Test Case 11	BC = connected	DC = connected	name = invalid value	/	/	/	Error
Test Case 12	BC = connected	DC = disconnected	/	/	/	/	Error
Test Case 13	BC = connected	DC = invalid connection	/	/	/	/	Error

Test Case 14	BC = connected	DC = null	/	/	/	/	Error
Test Case 15	BC = disconnected	/	/	/	/	/	Error
Test Case 16	BC = invalid connection	/	/	/	/	/	Error
Test Case 17	DC = null	/	/	/	/	/	Error

5.2 White Box Testing (using unit testing)

Testing the AI Features inside of our application

- **Banknotes Testing**

```
@app.route('/test')
def test_banknotes():
    response = bn.base64_pil('/test_img1.png')
    if response == '10Egp':
        return 1
    else:
        return 0
banknotes: 1
```

Figure 13: Represents the Banknote Test Function and response

- **Text Testing**

```
@app.route('/test')
def test_OCR():
  response =  base64_pil('/test_img1.png')
  if response == 'Ahmed':
     return 1
  else:
     return 0
OCR: 0
```

Figure 14: Represents the Text Test Function and response

- **Indoor Scene Testing**

```
@app.route('/test')
def test_clothes():
  response = sd.classify_place('/test_img1.png')
  if response == 'library':
     return 1
  else:
     return 0
1
```

Figure 15: Indoor Scene Test Function and response

- **Face Testing**

```
@app.route('/test')
def test_face():
  response = Fr.new_friend('/test_img1.png')
  if response == 'saved':
     return 1
  else:
     return 0
1
```

Figure 16: Face Test Function and response

- **Clothes Testing**

```
@app.route('/test')
def test_clothes():
    response = cd.clothes_predict('/test_img1.png')
    if response == 'backpack':
        return 1
    else:
        return 0
```

```
1
```

Figure 17: Cloth Test Function and response

- **Object Testing**

```
@app.route('/test')
def test_object():
    response = Obj.search_item('/test_img1.png','laptop')
    if response == '1 laptop':
        return 1
    else:
        return 0
```

```
1
```

Figure 18: Object Test Function and response

Wherefore we have achieved testing by little meaning and covered a good portion of cases of our client/user needs.

Chapter 6

Chapter 6: Software

6.1 AI Models:

6.1.1 Handwriting recognition:

We searched about which algorithm we would use in this model, in the first we found the dynamic programming, but we found in it many disadvantages after trying many models of it and doing many pieces of training.

Disadvantages as:

1. No general formation of Dynamic Program is available; every problem has to be solved in its own way.

2. Dividing the problem into sub problems and storing intermediate results consumes memory.

3. It is difficult to develop code using dynamic programming as opposed to greedy technique. Also different DP have different methods of solving them

So we decided to work by another algorithm. We use a NN for our task. It consists of convolutional NN (CNN) layers, recurrent NN (RNN) layers and a final Connectionist Temporal Classification (CTC) layer.

Because the advantages of Recurrent Neural Network (RNN):

1. RNN captures the sequential information present in the input data i.e. dependency between the words in the text while making predictions.

2. RNNs share the parameters across different time steps. This is popularly known as **Parameter Sharing**. This results in fewer parameters to train and decreases the computational cost.

-And the Figure 19 represented the Recurrent Neural Network

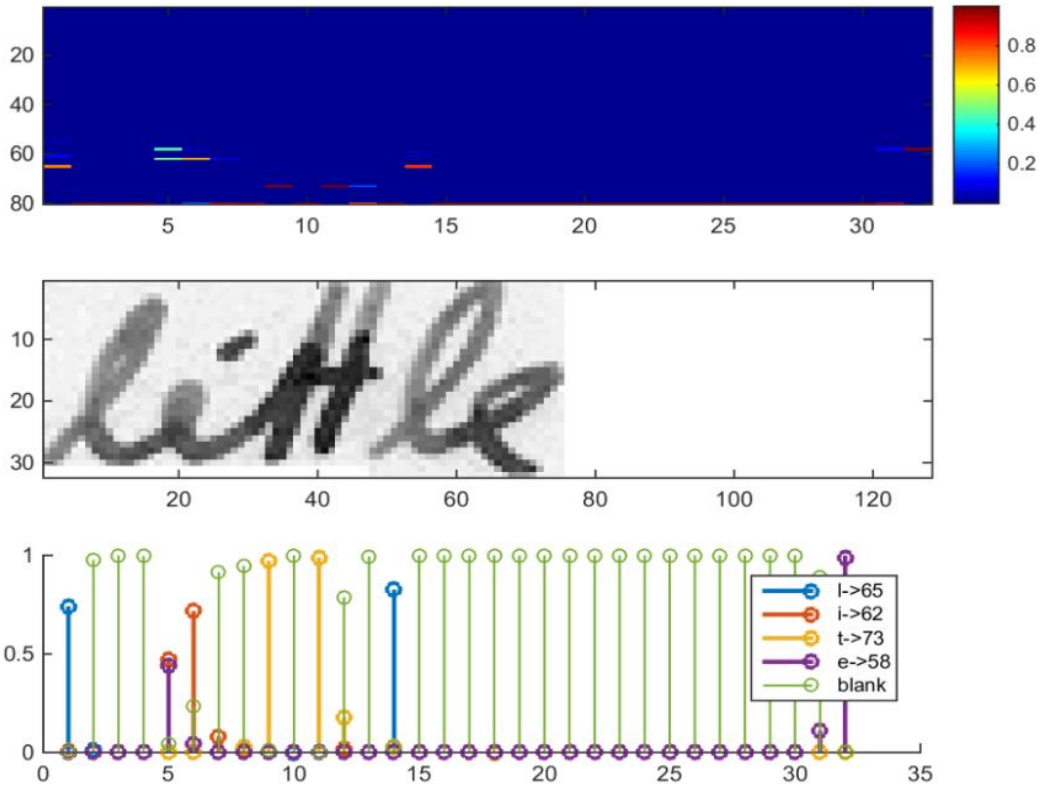


Figure 19: Recurrent Neural Network Layers

Because the advantages Convolution Neural Network (CNN**):**

1. CNN learns the filters automatically without mentioning it explicitly. These filters help in extracting the right and relevant features from the input data.

2. Captures the **spatial features** from an image. Spatial features refer to the arrangement of pixels and the relationship between them in an image. They help us in identifying the object accurately, the location of an object, as well as its relation with other objects in an image.

Input image: The model took the input image as jpg or png but it comes from the server as base64 so we change the format.

-The figures 20 represent handwriting recognition model output.

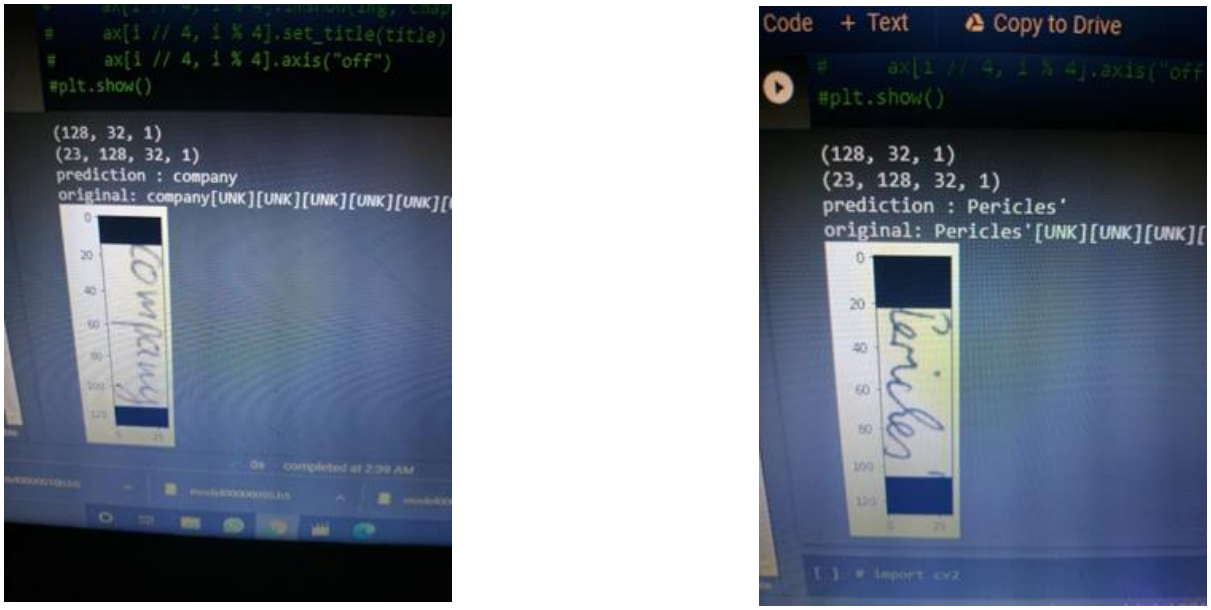


Figure 20: Handwriting Recognition Console Output

6.1.2 Text Recognition:

Python offers many libraries to do this task. There are several ways of doing this, including using libraries like Paddle. OCR can read text from scanned images. OCR operates in 2 steps such as text detection and text recognition. OCR is the technology that is used to convert the scanned image, paper document images captured by digital camera and extract the text from images which are automatically updated. The image is processed using Open-CV libraries. The Open-CV and Nano net OCR is the powerful technique in this field to extract the text from the image. To implement and verify the research, we use an android application. The process starts by taking the image and extracting the text then it is automatically updated. Text detection: The Text Detection module detects the text in the image for further recognition. The Efficient and an accurate Scene Text Detector algorithm is used for the text detection process.

With this feature we can provide a reader to the user that helps him/her with reading the normal texts (e.g. Text from a Book or from a Planner) also the model can recognize the numbers so the user may use this feature to know the prices which written on the product also he/she may use this feature to know the medicine name without any help just using the camera. As shown in Figure 21 sample of text recognition output.

Figure 21: Text Recognition Console output

6.1.3 Face Recognition:

Why use face recognition for blind person?

- In order to make the Blind person more and more independent
- helping him find his friends using only the mobile phone camera
- avoiding the embarrassment asking strangers

How to use face recognition?

- Using his mobile phone and our Face-recognition feature that: -

- Recognize and manipulate faces from Python
- Is built using Dlib's state-of-the-art face recognition
- Is built with deep learning.
- Has an accuracy of 99.38%.
- Using Labeled Faces in the Wild Data-set.

-In the following Figure 22 NN explaining the how the model extracts the face feature and subtract the reference face from the test face

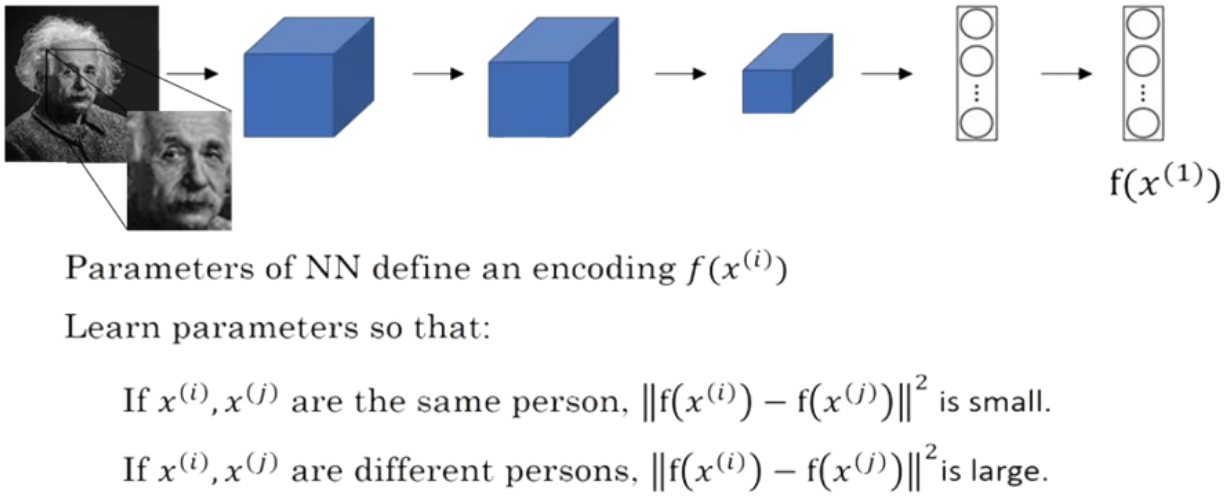


Figure 22: Face extraction features using Neural Networks

-In Figure 23 here is a customized real-time implementation of the face-recognition model

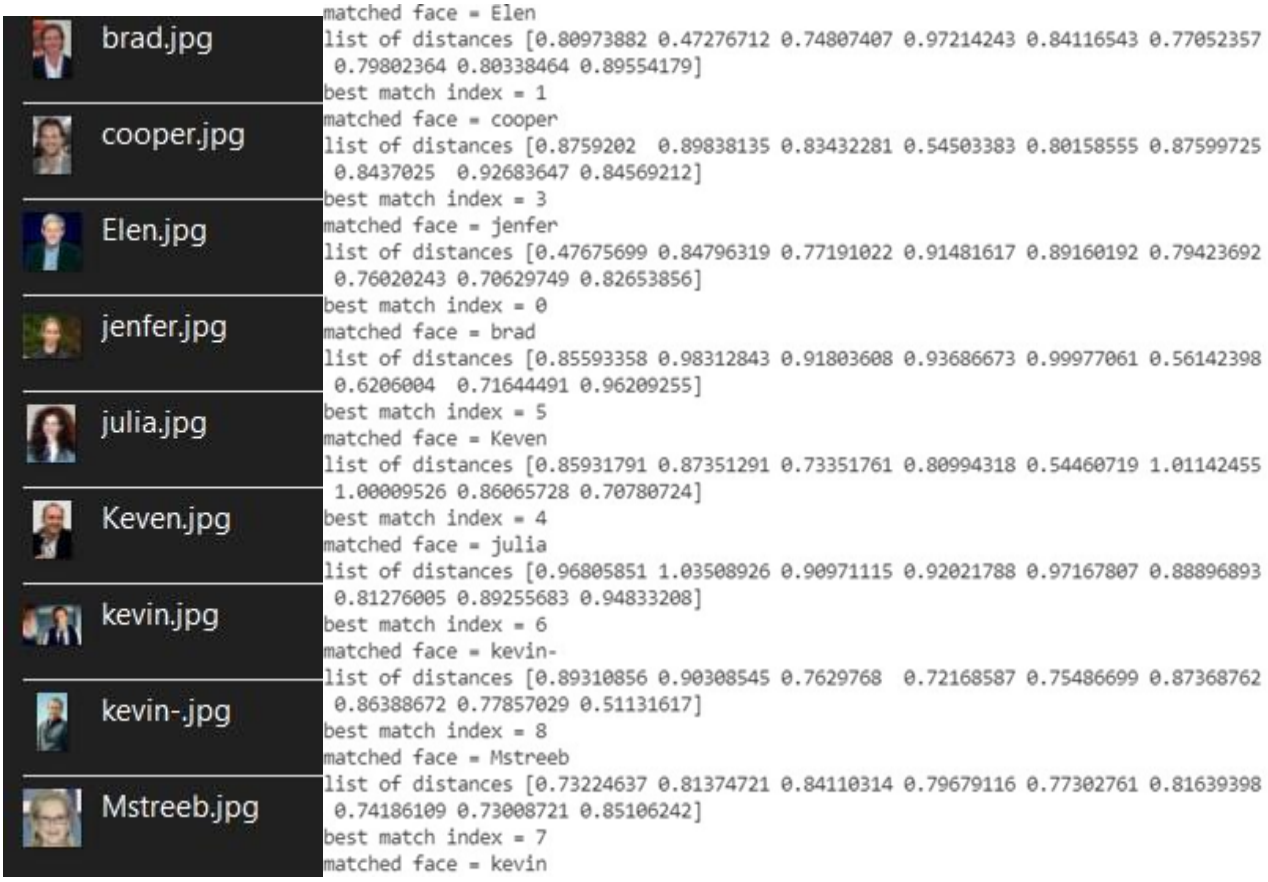


Figure 23: Real Time Implementation of Face Recognition 1

The Figure 24 shows the real output of using the above data as a reference for the faces.

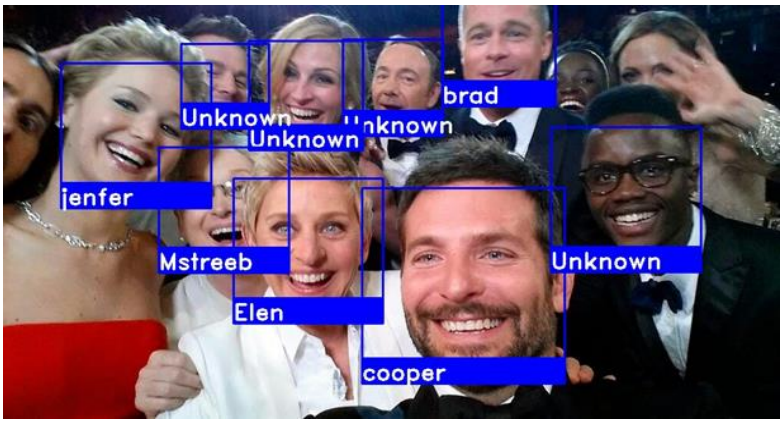


Figure 24: Real Time Implementation of Face Recognition 3

The figure 25 represented the face recognition model test.

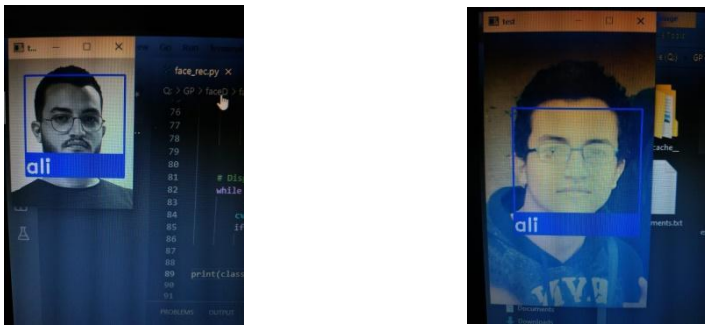


Figure 25: Face Recognition Developer Test

The model recognized the same person however the difference in the look and Age. But we don't need to draw a rectangle around the face for our users so we decided to remove all the drawing and return only the results as **a text** that will be converted into a speech using TTS.

Also, our model is communicating with the application through the server that will send a picture that is converted into a **base64** form so we decode this base64 form into a normal '.jpg' form then we test the picture then we give it to the model and send back the result as a text that will be read by TTS (text to speech) feature.

According to the use of this module, the user will need to go through some scenarios which we will discuss through the following flow charts

(Figures26, 27, 28): -

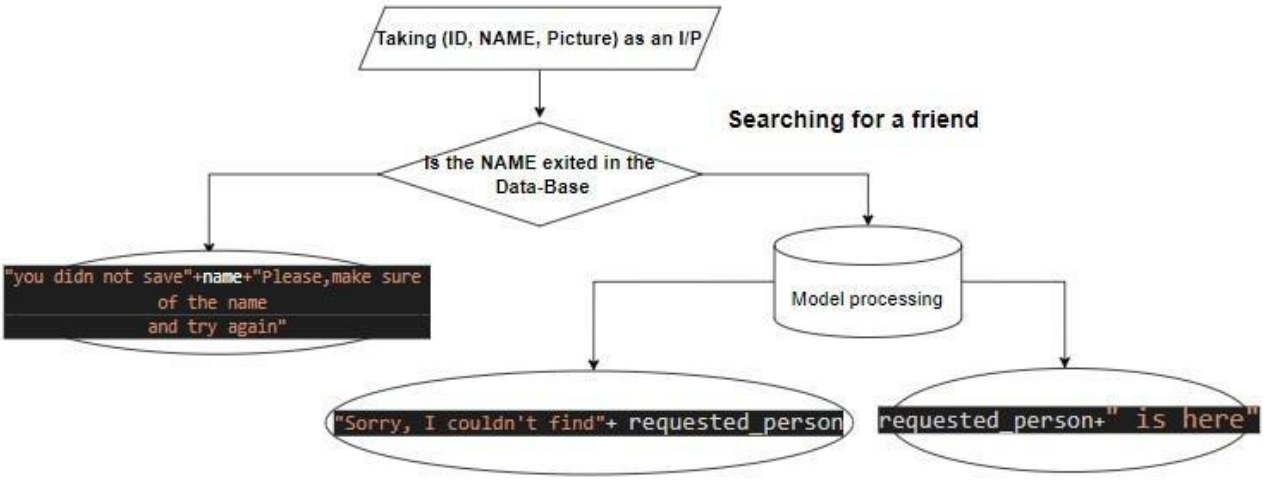


Figure 26: Face Recognition Flow Chart 1

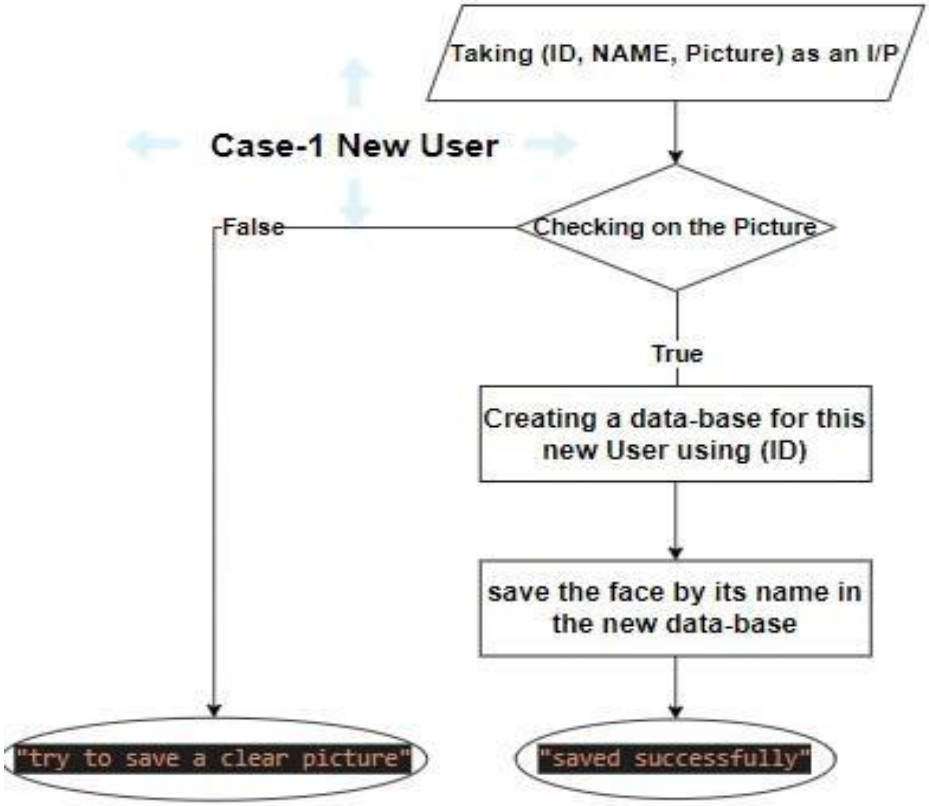


Figure 27: Face Recognition Flow Chart 2

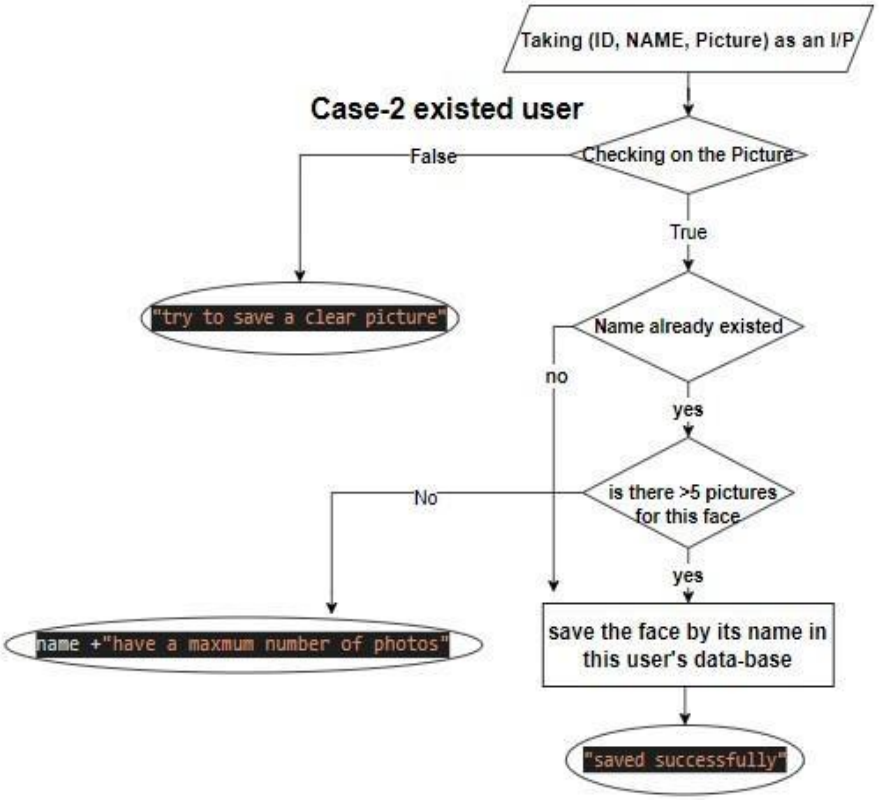


Figure 28: Face Recognition Flow Chart 3

The figure 29 represent the hierarchy of the database of the saved faces for the different users based on the User ID

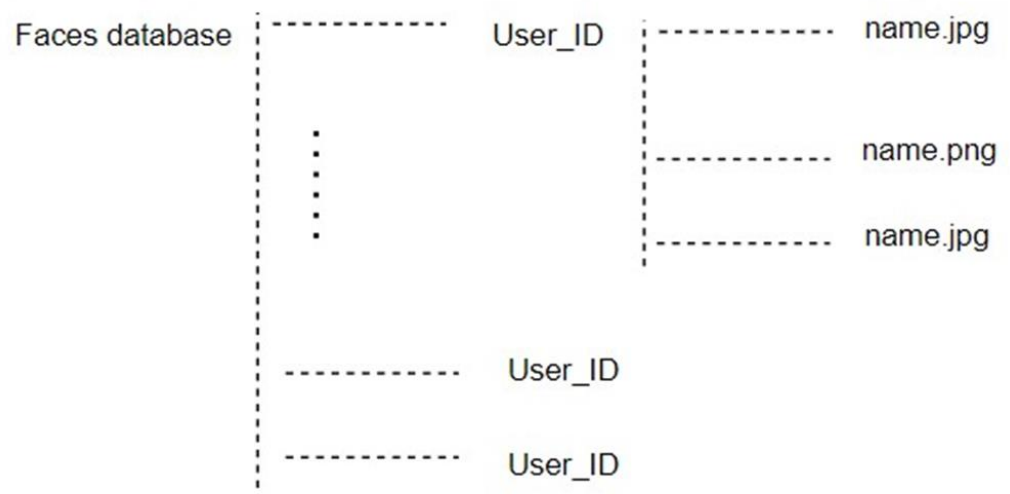


Figure 29: Face Recognition Database

6.1.4 Indoor Scene Recognition:

Why use indoor recognition for the blind person?

The Basic Idea sets behind the exploring we do every day using only our eyes. Blind people however are just exploring part of this world using every other sense but not the vision. It may be practically useful sometimes for them to use all other senses, nevertheless giving the chance to a blind person to be capable of creating a visual picture in his/her mind using only his/her voice or finger is priceless. The 39-classes version of the Indoor scene recognition Deep Learning model enables blind people to create mind-simulated images allowing them to explore new experiences and picture the reality of their surroundings.

How using indoor recognition?

Using Deep learning we trained a Classification model with 39-classes using the **Indoor Scene Recognition, CVPR 09** Dataset from MIT.

The dataset contains **67** Indoor categories, and a total of 15620 images. The number of images varies across categories, but there are at least 100 images per category. All images are in jpg format.

Our first trial was for 67 classes using the dataset **as it is** but we faced a huge problem called The **over Fitting**.

What is over fitting?

-Over fitting is simply when there is a huge gap between test/validation accuracy and the training accuracy.

After the first training the validation accuracy didn't break the 55% on the other hand the training accuracy was + 90% which is a very huge gap.

How did we go through the solution of this problem?

Knowing that the reasons of the over fitting: -

Reason_1: The dataset does not provide enough data to improve the performance of the model

Solution: 1st we did a stronger data augmentation-showed in the Figure 30 in order to increase the amount of data that the model will use during the training process.

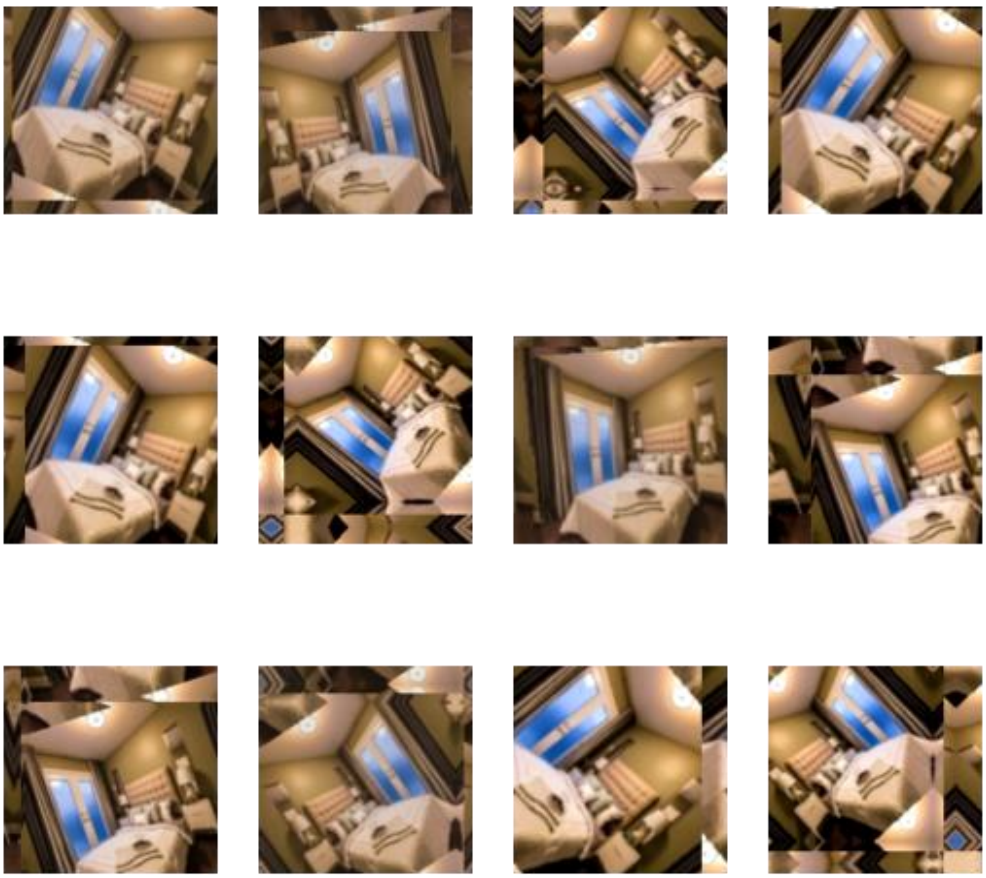


Figure 30: Indoor Scene Recognition Data Augmentation

As we can see the data Augmentation is a method of duplicating the same picture using a random change of **flip, rotation, zoom and contrast**

But that wasn't enough, because the data we are using is still not enough as we need to train the model on 67-different classes and 67 is not a small number

So we started to make some changes on the original dataset (**CVPR 09**) the original dataset was **67** Indoor categories, and a total of 15620 images

So we removed the unwanted categories and the kind of duplicated categories and although we added new categories (e.g. mosque inside) at the end we had only **39** categories but decreasing the number of categories is not enough for sure, so we collected around +11K more picture then we cleaned those pictures to be only JPG.

Images then we filtered the data from all the bad paths images in the end we had a new dataset with **26589 images belonging to 39 classes which names are :-**

[auditorium, bakery , bathroom, bedroom, bookstore, children room, classroom, church inside, clothing store, concert hall, corridor, deli, dining room, fastfood_restaurant, florist, grocery store, gym, hair salon, hospital room, inside subway, kindergarten, kitchen, library, living room, lobby, mall, meeting room, nursery, office, pantry, pool inside, restaurant, staircase, shoe shop, toy store, waiting room, warehouse, laundry, inside mosque]

For the first trial the validation accuracy didn't break the 55% on the other hand the training accuracy was + 80% which is a very huge gap and a low accuracy too.

After the new dataset we tried again, and we got a noticeable progress as shown in the following Figure 31.

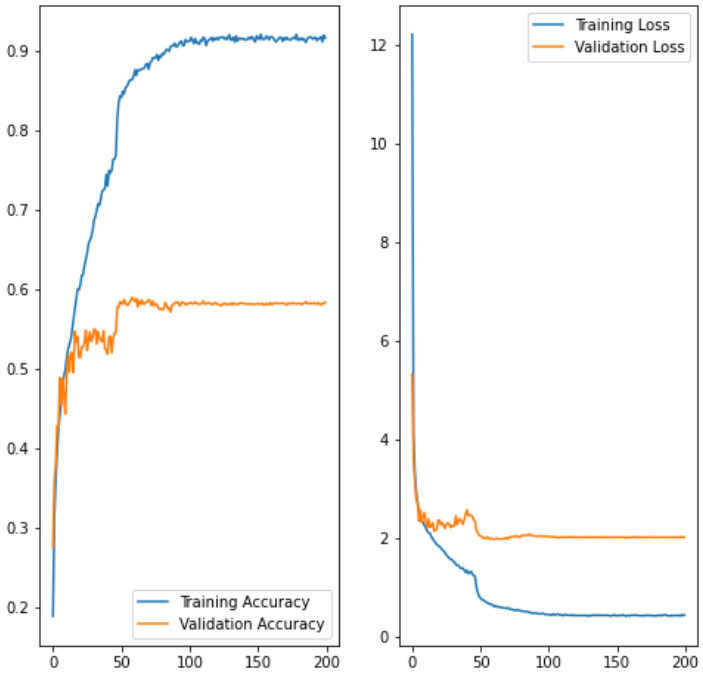


Figure 31: Over fitting Due To Minimum Data

We got + 60% validation accuracy and +95% training accuracy but still there is a gap between them…

Reason_2: **Dropout rate**

Dropout is a technique for addressing the problem of over fitting. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from co-adapting too much. As shown in Figure 32.

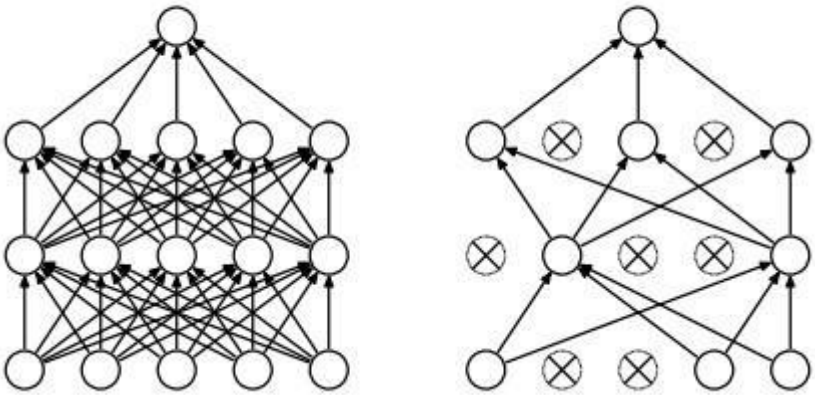


Figure 32: Indoor Scene Recognition Dropout Rate explanation

After some research we found out that the most common dropout ratios are between 0.5 and 0.8 and we were using a 0.25 dropout ratio so we started to train the model using 0.5, 0.75, 0.85, 0.8 and the best we could get so far is **0.8** with the following results in Figure 33.

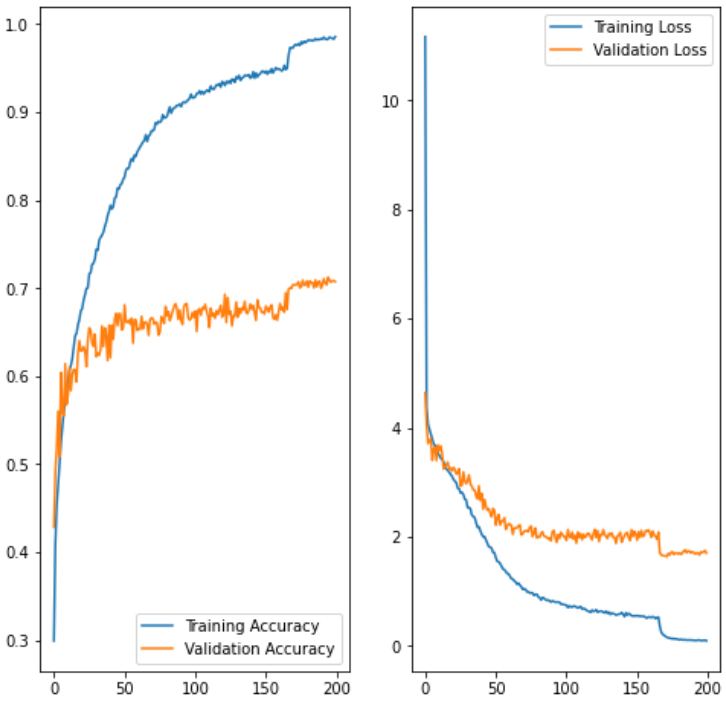


Figure 33: Over fitting Due To Dropout Rate

At the end we got Training accuracy 98.2% and a Validation accuracy 72.315%

Results of the last Model as shown in figure 34.

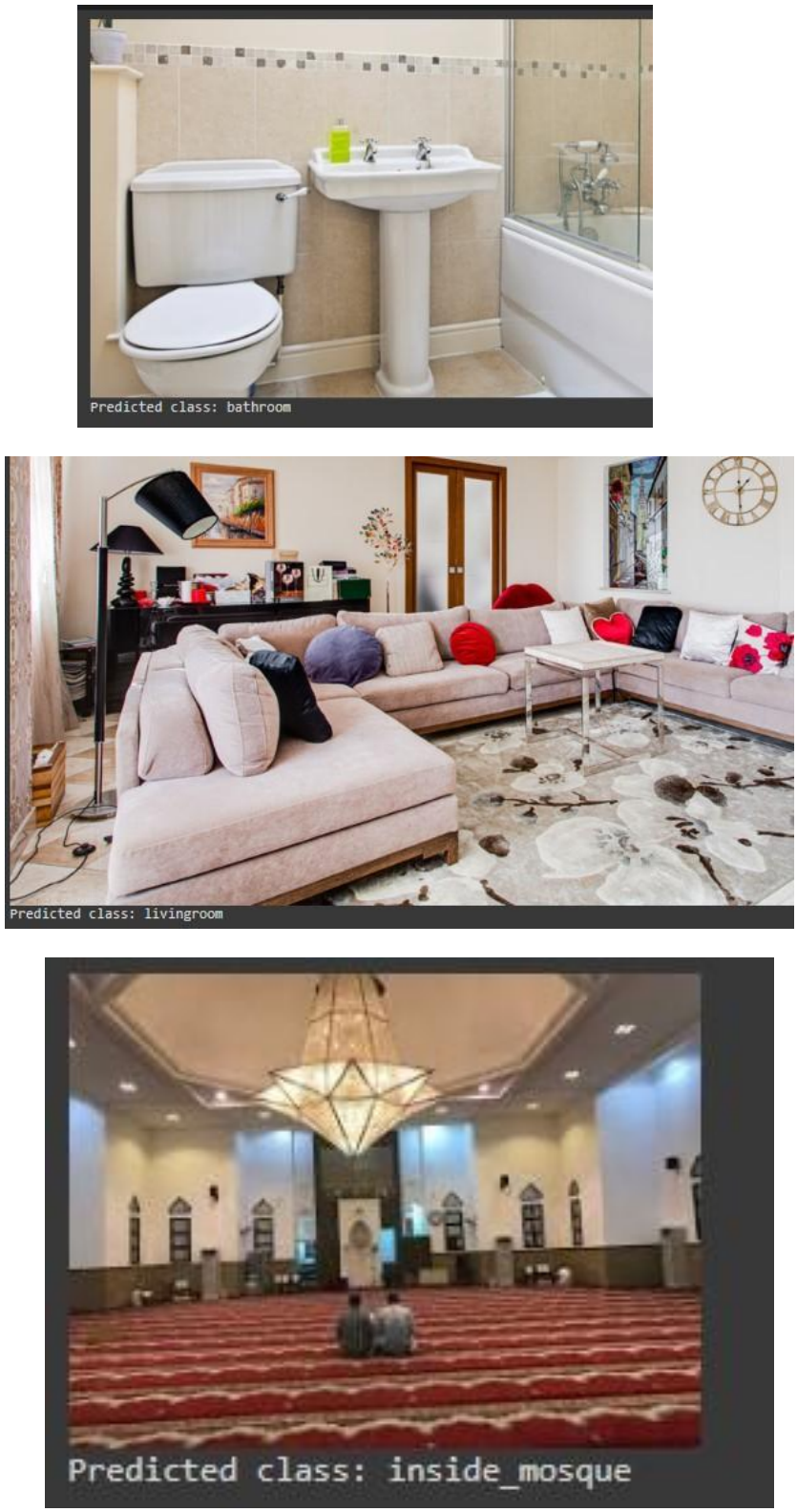


Figure 34: Indoor Scene Recognition Console Output

6.1.5 Object detection:

-Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Well-researched domains of object detection include face detection and Pedestrian Detection. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance.

- State of the art computer vision systems have involved a range of object detection models that use convolutional neural networks in their working. The deployment of these models like YOLO, SSD, R-FCN, R-CNN, etc.

First, we are trying with YOLO but we discovered that SSD is better than YOLO in many points as:

1. We have found SSDs much easier to train and their performance in terms of accuracy almost always outperforms YOLO (at least for the datasets we have worked with).

2. YOLO may have excellent results on the COCO dataset; however, I have not found that same level of accuracy for my own tasks.

SSD:

Is a popular algorithm in object detection? It’s generally faster than Faster RCNN.

SSD Mobile Net Architecture:

The SSD architecture is a single convolution network that learns to predict bounding box locations and classify these locations in one pass. Hence, SSD can be trained end-to-end. The SSD network consists of base architecture (Mobile Net in this case) followed by several convolution layers:

-And the SSD Based Detection with Mobile Net is represented in figure 35.

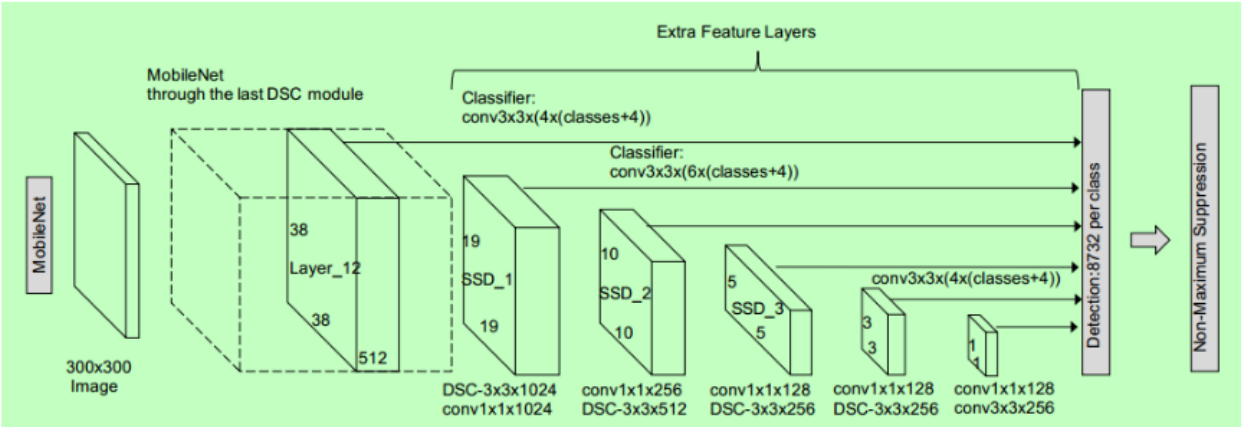


Figure 35: SSD Based Detection with Mobile Net

By using SSD, we only need to **take one single shot to detect multiple objects within the image**, while regional proposal network (RPN) based approaches such as R-CNN series that need two shots, one for generating region proposals, one for detecting the object of each proposal. Thus, SSD is much faster compared with two-shot RPN-based approaches.

Object detection operation:

-This model can detect more than 90 objects. SSD works by converting discrete output spaces for bounding boxes into sets of default boxes for different aspect ratios and for every feature map location. During predictions, the model generates the scores in each default box for every object detected and scales the default box to fit the object shape. SSD is simpler than other networks as it performs all computations in a single network.

-The figure 36 discusses the object detection.

Figure 36: Object Detection

Model operation:

-How the application works for blind people with the images and boxes that detect the object and show the name and the scores that cannot help the blind. So, the AI part can convert these boxes into text that will convert into voice.

The blind can search for any object in object detection mode and in the code will convert any capital character into the lowercase to avoid any error, the output will be the number of the object searched for and its name.

For example, if the blind search for a “chair” the output will be “4 chairs” as text that the application will convert to voice.

What will happen if the blind searches for any object and this object is not in the image that he searched in it the application will reply to him with “Not found!” as text.

6.1.6 Clothes detection:

This model is created by Detecto.

Detecto: is a Python package that allows you to build fully-functioning computer vision and object detection models. Detecto is an open-source library for computer vision programming that helps us in fitting state-of-the-art computer vision and object detection models into our image data. One of the great things about this package is we can fit these models using very few lines of code. In the examples, they have shown us how we can fit these models using only four or five lines of code.

Inference on still images and videos, transfer learning on custom datasets, and serialization of models to files are just a few of Detecto's features. Detecto is also built on top of PyTorch, allowing an easy transfer of models between the two libraries. The power of Detecto comes from its simplicity and ease of use. Creating and running a pre-trained Faster R-CNN ResNet-50 FPN.

Dataset:

-The model can detect 10 types of clothes:

['Backpack' , 'Dress' , 'Footwear' , 'Handbag' , 'Jacket' , 'Jeans' , 'Shirt' , 'Skirt' , 'Suit' , 'Watch'].

-For training, it trained at 10,000 images for training and 5,000 images for validation.

The operation:

-How the application works for blind people with the images and boxes that detect the object and show the name and the scores that cannot help the blind. So, the AI part can convert these boxes into text that will convert into voice.

The blind can search for any type of clothes in clothes detection mode and the code will capitalize the input to avoid any error, the output will be the number of the type of clothes searched for and its name.

What will happen if the blind searches for any type of clothes and this type is not in the image that he searched in it? The application will reply to him with “Sorry no clothes here, try again!” as text and converted to voice.

-Figure 37 describes when searching for “Suit” in this image:

The output will be “1 Suit”.

-And when searching for “Shirt” in this image:

The output will be “1 Shirt”.

Figure 37: Cloth Recognition Console Output 1

-Figure 38 describes when searching for “Backpack” in this image:

The output will be “1 Backpack”.

Figure 38: Cloth Recognition Console Output 2

- All of this text the application will convert to voice.

6.1.7 Color Classification:

-The model can classify White, Black, Red, Green, Blue, Orange, Yellow, and Violet.

In this model, we used the K-Nearest Neighbors Machine Learning Classifier (KNN) which is widely used in neural networks and machine learning algorithms. KNN algorithm is a supervised classification algorithm that needs labeled data to train on. By Using color histogram feature extraction, which is one of the image processing techniques, the features that distinguish these colors are determined. These features increase the effectiveness of the KNN classifier

-Figure 39 represented the color detection.

Figure 39: Color Detection

. KNN is trained by using R, G, and B Color Histogram values. The general steps involved are stated as represented in figure 40

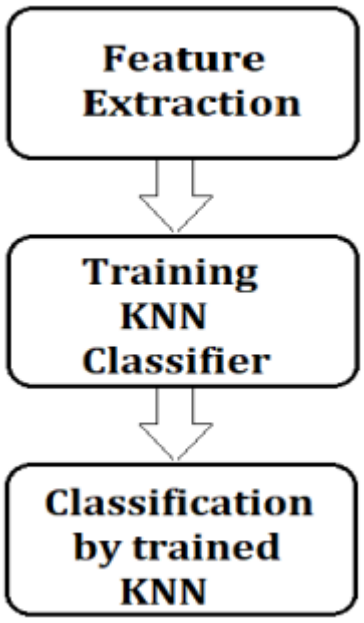


Figure 40: KNN Approach

Feature Extraction:

-Color Histogram represents the color distribution of an image. In digital images Color histogram refers to the number of pixels that have colors in a list of color ranges that distance the image's color space, which is the set of all possible color values. RGB color histogram of an input image can be obtained. Bin number of a histogram is used with the highest value of pixel count for RGB as attributes so we can get the dominant RGB values for making feature vectors for training.

The RGB values for each training image are obtained by color Histogram and are labeled because the KNN classifier is a supervised learning algorithm.

Training KNN classifier:

-KNN classifier algorithm is trained using RGB Color Histogram values.

Classification using trained:

-KNN An algorithm that executes classification in a concrete execution is known as a classifier. The word "classifier" refers to the mathematical function, which is executed using a classification algorithm that plots given data to a group. KNN algorithm stores all the cases obtained and the classification of new cases is done on the basis of the similarity measure.

-Figure 41 represent KNN classification process.

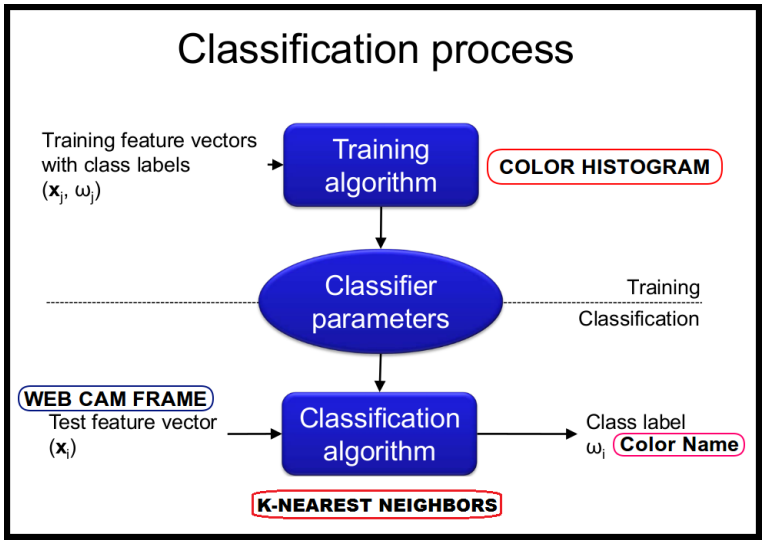


Figure 41: KNN Classification Process

6.1.8 Banknotes Recognition:

-The banknotes recognition mode is one of the important modes made in this application for protecting the blind from scam methods that unfortunately happen to the blind. The fraud people exploit the obstruction.

So, this mode in the application will protect the blind from these methods.

-The model of banknotes is created by the Detecto python package.

Dataset:

-The dataset used in this model is Egyptian banknotes.

The model can detect:

[‘5 EGP’, ‘10 EGP’, ‘20 EGP’, ‘50 EGP’, ‘100 EGP’, ‘200 EGP’].

For training, it trained at 6,000 images for training and 3,000 images for validation.

The operation:

How the application works for blind people with the images and boxes that detect the banknote and show the name and the scores that cannot help the blind. So, the AI part can convert these boxes into text that will convert into voice.

What will happen if the blind take a photo without any banknotes? The application will reply to him with “Sorry no banknotes here, Please try again!” as text and converted to voice.

-Figure 42 represented the banknotes recognition.

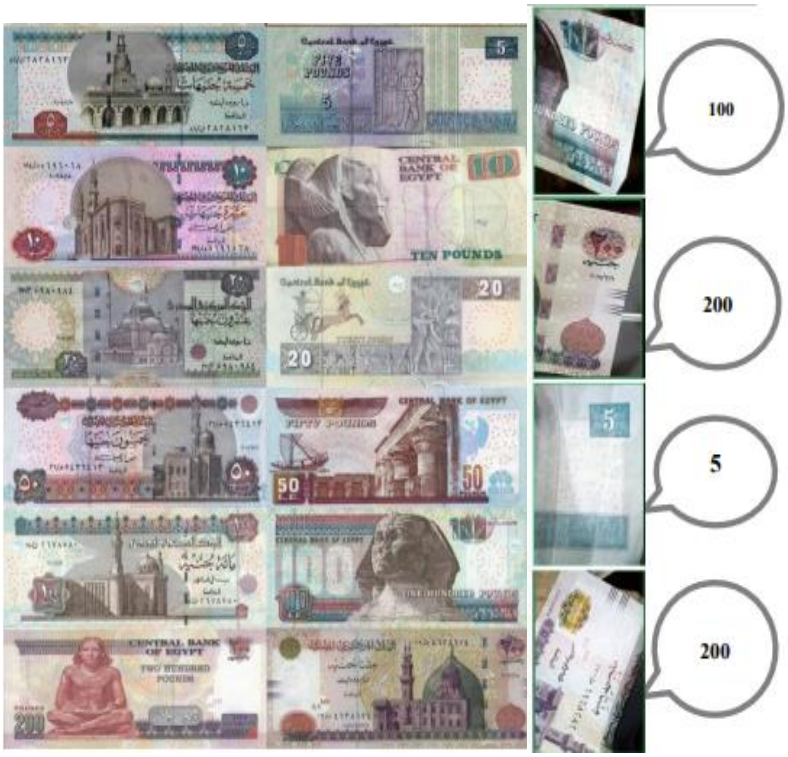


Figure 42: Banknotes Recognition

6.2 Server:

The server is responsible for the communication between the Ai models and the mobile application, as the mobile application captures the picture and then the picture is sent to the server that uses the model to get response that is sent back to the client side, therefore the server is an integral part of the system.

The language used to implement the server is python using framework flask.

What is Flask?

Flask is a micro framework offering all the basic features of a web application. It's easy to get started on and expand with Python's wealth of libraries and extensions.

On its own, the framework's capabilities are limited and extra functionalities need to be added with supplemental technologies. In many ways, it takes an opposite approach to that of Django, as it enables developers to change and tailor it to their particular needs at will.

With Flask, diversifying the structure of your software project with micro frameworks is effortless. Flexibility is the core feature of this open-source framework.

What is Flask used for?

Well, Flask may be the right choice when you know there will be more technological decisions to make on the way, and you don't want to be limited by the decisions you would've already made.

If that sounds familiar, you're probably considering a whole bunch of solutions, trying to find the best one and go with it. That's perfectly fine. Flask gives you a wide variety of options when you have little to no expectations.

In short, Flask is a great fit in three main instances:

1. Experimentation with the architecture, libraries, and newest technologies.
2. Unknown number of highly specialized (micro) services.
3. Multitude of small, disparate features.

Conversely, Flask also works really well with web apps we know for a fact will be divided into many smaller services that won't require that much code to accomplish their goals. Rather, they have a simple goal supported by simple implementation. An example here would be web applications that fall under larger systems used in DevOps, like monitoring or large-scale logging solutions.

Why use Flask?

Django is opinionated, monolithic, and fixed, making customization more difficult. For instance, using a different ORM framework in Django is possible, but it involves more non-standard work from your developers.

When you're not sure which libraries or frameworks will work best for your software product, we recommend that you go with Flask.

Virtually every Flask project out there has its own unique "internal tech stack" of libraries, frameworks, etc. It's something of a free-for-all.

Because of this, Flask is widely known to facilitate experimentation to a high degree. It's really easy and admittedly very fun for your developers, which in turn means lower turnover for you.

Therefore in conclusion the Advantages of Flask-based systems are :

- Higher flexibility.
- Higher compatibility with latest technologies.
- High scalability for simple web applications.
- Technical experimentation.
- Customization.
- Slightly higher framework performance.
- Easier to use for simple cases.
- Smaller size of the code base.

We found two options for the server that can be used:

1- Rest API.

2- Socket Server.

Let's discuss both in detail:

6.2.1 Rest API

- Rest API is good for sending requests and receiving responses. But it is one way. The user must send a request to receive a response. The server can't send data to the user directly.

6.2.2 Socket

- Socket is the most popular server that is used in real time applications.
- Implementing a socket server has some difficulties, so we looked for a simple package that would make it easier to implement. That package is called Socket.IO

6.2.3 Socket.IO

Performance: In most cases, the connection will be established with Web Socket, providing a low-overhead communication channel between the server and the client.

Reliable: Rest assured! In case the Web Socket connection is not possible, it will fall back to HTTP long-polling. And if the connection is lost, the client will automatically try to reconnect.

Scalable: Scale to multiple servers and send events to all connected clients with ease.

Socket.IO can be implemented by many languages such as Node.js, Dart, C++, Swift and Python. The most suitable choice as discussed here is to implement it by python-flask because we have our AI models that are implemented by python.

-The figure 43 represent the socket figure console.

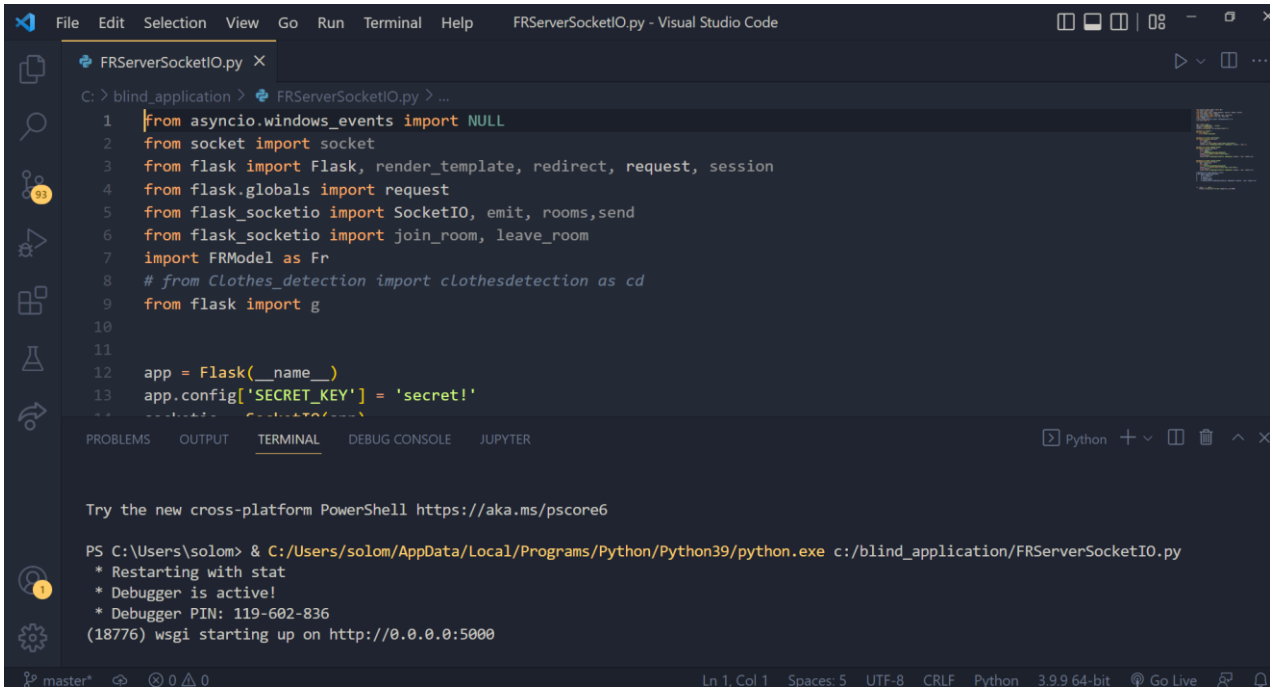


Figure 43: Socket Server Console

6.3 Mobile app:

6.3.1 STT (Speech to text):

Active (handwriting or text) detection mode:

.Change mode in server to send directly to the suitable model.

.TTS (Text to speech): Receive response.

The blind can search by his voice for any object he wants and the application will reply to him by the number of objects that he searched for and the name of it.

Speech to text (STT)

The blind cannot see the mobile screen, so something that the blind can receive the response must be chosen. Blind people can hear, so we must use STT to respond to the blind.

When we tried to integrate our application with STT we faced the same problems as TTS. So, we did the same solution. We used the built-in STT to gain the same advantages.

Advantages of using mobile build in STT:

1- It is local so the response will be real time.
2- No need to send a voice to the server and analyze it, then the server responds with something to take an action.
3- Sending a string will be much faster than sending an audio file.

6.3.2 Text to speech (TTS):

-Blinds can't deal with the mobile using their hand, so we must choose something the blind can do with ease, which is speech. But how can we use his speech to do some action in the application? The answer is using Speech-To-Text (STT). Let's discuss it in detail.

-When we tried to integrate our application with STT we found many AI techniques, but all of them will work on the server. Which will overload the server and its response will be late. So, we thought that it is more efficient to use the mobile built-in STT. The mobile nowadays has a very efficient STT model.

Advantages of using mobile build in TTS:

1- It is local so the response will be real time.
2- No need to convert the response to speech and send it from server to blind.
3- Sending a string will be much faster than sending an audio file, so it is
4- Better to send a string from the server and convert it to speech on the mobile.

6.4 Video call

No one can predict every need of blinds. Therefore, no one can program applications that can help the blind in everything. They need help from people, the application can help them in many aspects in their life. But what if the blind need something that does not exist in the application? Volunteer's video call is the solution. Blinds can look for a volunteer who can help them in something that does not exist in the application such as: checking how long time for the washing machine left to stop. But here is a question. What if there is no strange volunteer to help? The blind can ask the application to call someone from his family who usually helps him.

6.4.1 Video call implementation:

We looked for various techniques that can help us to implement video call. We find two options:

1- Client-Server.
2- P2P (Peer-to-peer).

Both have their advantages and disadvantages. Let's discuss every technique in detail.

Client-Server

- **Disadvantages**
  - Video call over Client-Server needs a supercomputer which has a high level of performance and therefore expensive cost.
  - Limited scale because even the highest computer performance in the world has its limitations.

- If all the clients simultaneously request data from the server, it may get overloaded. This may lead to congestion in the network.
- In client server computing, a server is a central node that services many client nodes.

- **Advantages**
  - Certainly, no technique has disadvantages only, but the advantage of the client-server is not useful for video call. So, no need to mention.

Peer-to-Peer:

Advantages:

- Clients collectively use their resources and communicate with each other.
- All the Clients are equal and share data with each other directly.
- The overall cost of building and maintaining a peer-to-peer network is relatively inexpensive. The setup cost has been greatly reduced since there is no central configuration.
- P2P networking has one of the best scalability features. Even if there are extra clients added, the performance of the network will remain the same.

Disadvantages:

- Hub or switch needed for the connection.

By comparing both systems we realized that the P2P is more suitable for video calls. We could solve the disadvantage of the P2P by setting up a small server that helps users exchange their connection data. After that we looked for peer-to-peer that is suitable for our application. We found WEBRTC.

6.4.2 WEBRTC

With WebRTC, we can add real-time communication capabilities to our application that works on top of an open standard. It supports video, voice, and generic data to be sent between peers, allowing us to build powerful voice-and video-communication solutions. The technology is available on all modern browsers as well as on native clients for all major platforms. The technologies behind WebRTC are implemented as an open web standard and available as regular JavaScript APIs in all major browsers. For native clients, like Android and IOS applications, a library is available that provides the same functionality. The WebRTC project is supported by Apple, Google, Microsoft, and Mozilla, amongst others.

As we noticed we need servers that help users exchange their data for the connection. That server will be implemented using flask-socketIO as discussed previously.

6.4.3 Signaling Server

WebRTC can't create connections without some sort of server in the middle. This server is called the **signaling server**. It's any sort of channel of communication to exchange information before setting up a connection

Peer A which in our case is the blind who will be the initiator of the connection, will create an Offer. They will then send this offer to Peer B that is the volunteer using the chosen signal channel. Peer B will receive the Offer from the signal channel and if he is available and creates an Answer. They will then send this back to Peer A along the signal channel.

So what is the data that is exchanged between peer A and peer B to establish connection?

6.4.4 Session Description Protocol (SDP):

The SDP is an important part of the WebRTC. It is a protocol that is intended to describe media communication sessions. It does not deliver the media data but is used for negotiation between peers of various audio and video codecs, network topologies, and other device information. It also needs to be easily transportable. Simply put we need a string-based profile with all the information about the user's device. This is where SDP comes in.

The SDP is a well-known method of establishing media connections as it appeared in the late 90s. It has been used in a vast amount of other types of applications before WebRTC like phone and text-based chatting.

6.4.5 ICE-candidate:

The RTCIceCandidate interface—part of the WebRTC API—represents a candidate Interactive Connectivity Establishment (ICE) configuration which may be used to establish an RTCPeerConnection . An ICE candidate describes the protocols and routing needed for WebRTC to be able to communicate with a remote device.When starting a WebRTC peer connection, typically a number of candidates are proposed by each end of the connection, until they mutually agree upon one which describes the connection they decide will be best. WebRTC then uses that candidate's details to initiate the connection.

6.4.6 STUN server

The scheme above works perfectly in a local area network where each peer has its own IP address and there are no firewalls and routers between them.

But when it goes to the internet, peers can be hidden behind NAT and do not have any information about the external addresses of each other.

-Figure 44 represent Web RTC using signaling server.

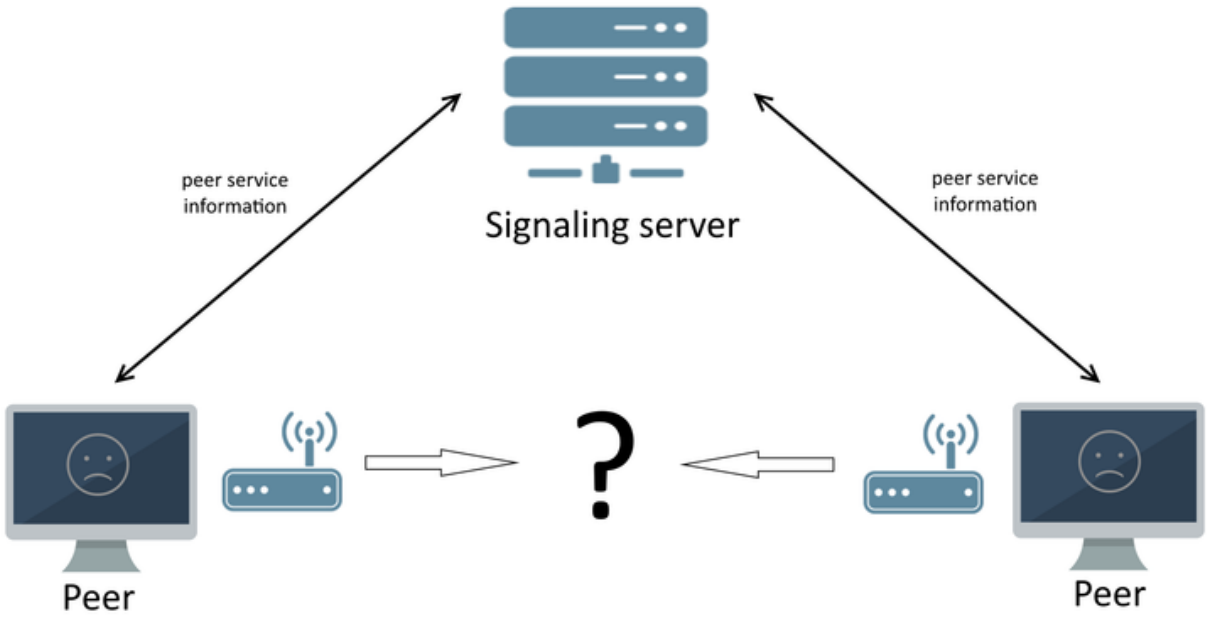


Figure 44: WebRTC Using Signaling Server

That's the point when we need a **STUN server**. It allows to detect peers public network addresses and establish a peer-to-peer connection behind a NAT.

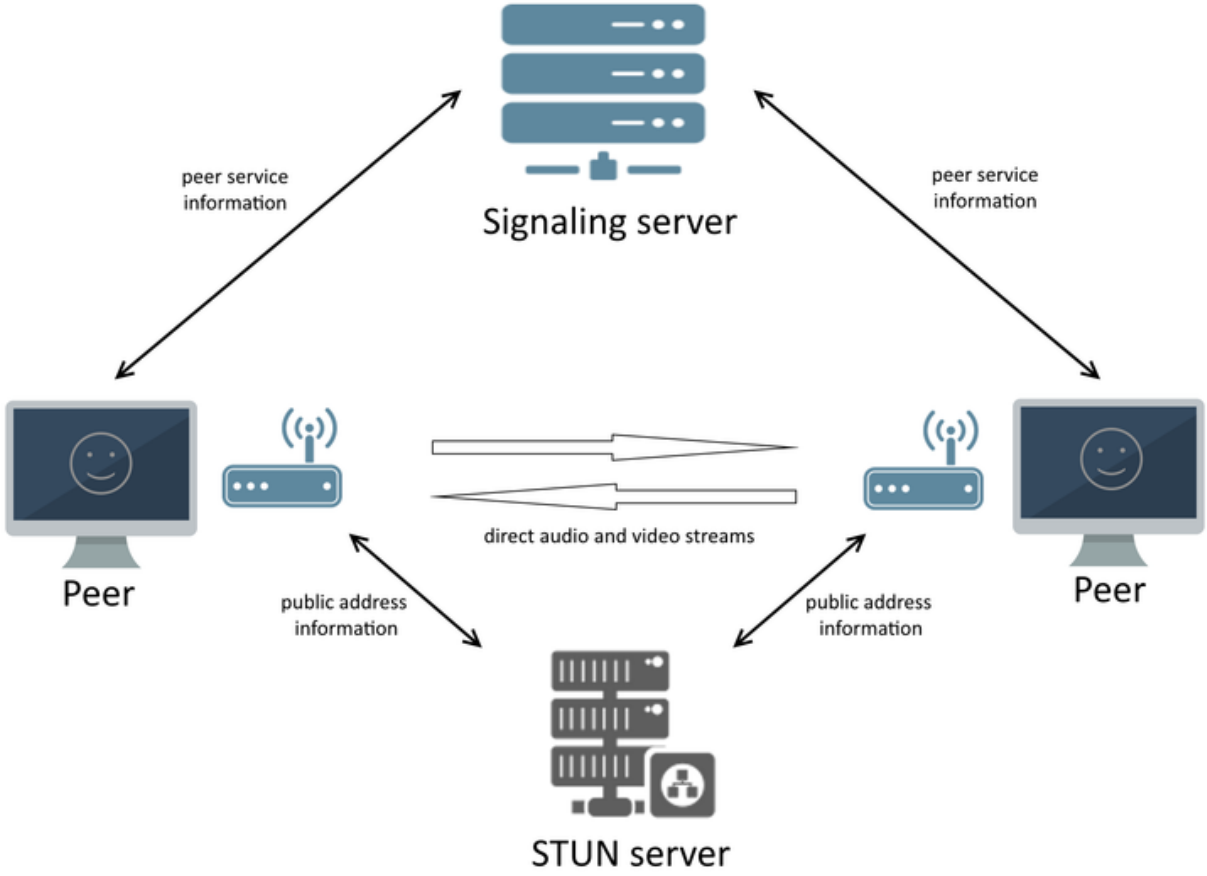


Figure 45: WebRTC Using Signaling Server and STUN Server

NOTE: The traffic and calculation load of the STUN server is relatively low, so you can use a public one or deploy your own.

6.4.7 TURN Server:

This scheme works fine in most cases. But there is another problem that can prevent direct peer-to-peer transmission: a firewall. It can be placed at any point of the network and cut the direct WebRTC traffic.

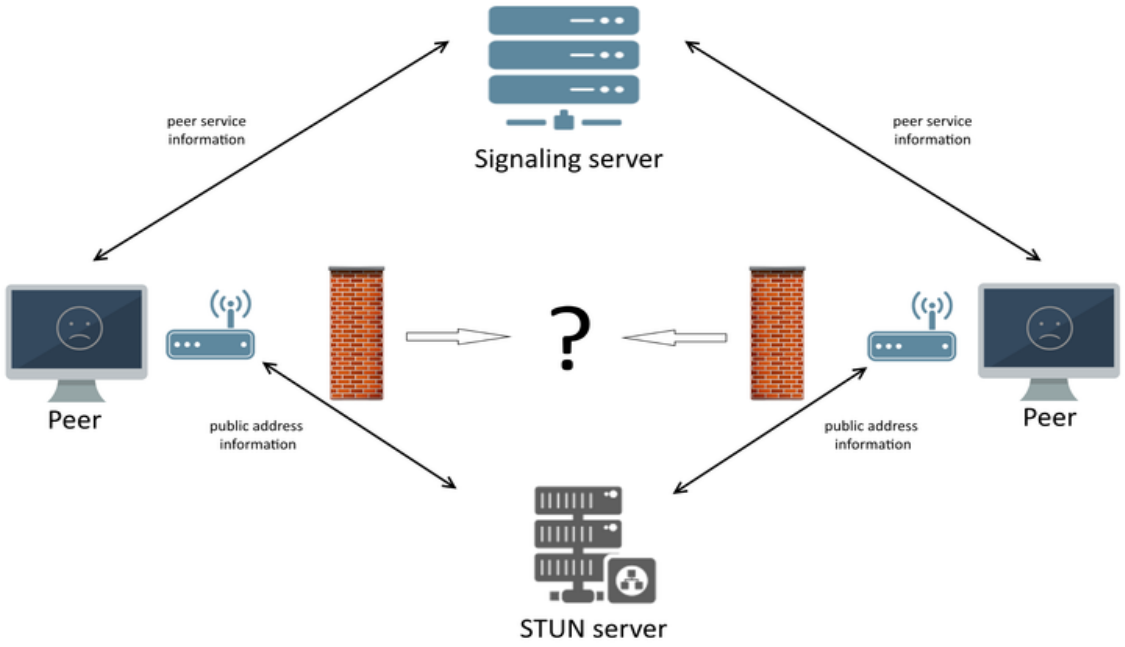


Figure 46: WebRTC Real Time Firewall Problem

A way to solve this problem is to use a **TURN server**. It has a public address, so both peers can interact with the TURN server even behind firewalls. So when no direct peer-to-peer connection is available, the TURN server transmits audio/video streams of both peers just like a common media server.

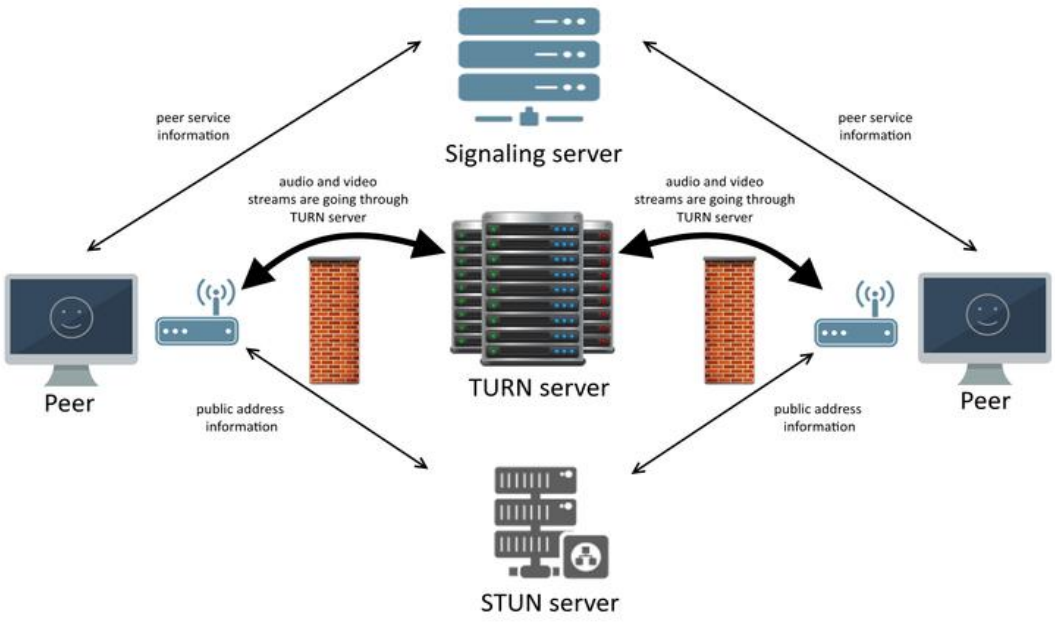


Figure 47: WebRTC Using Signaling, TURN and STUN Servers

NOTE: As TURN server transmits the media streams between peers it consumes a lot of traffic and requires a lot of calculation power, especially in a case of multiple peers processing. We strongly recommend you to have a dedicated server for this task and make sure that the traffic bandwidth is big enough to handle this task.

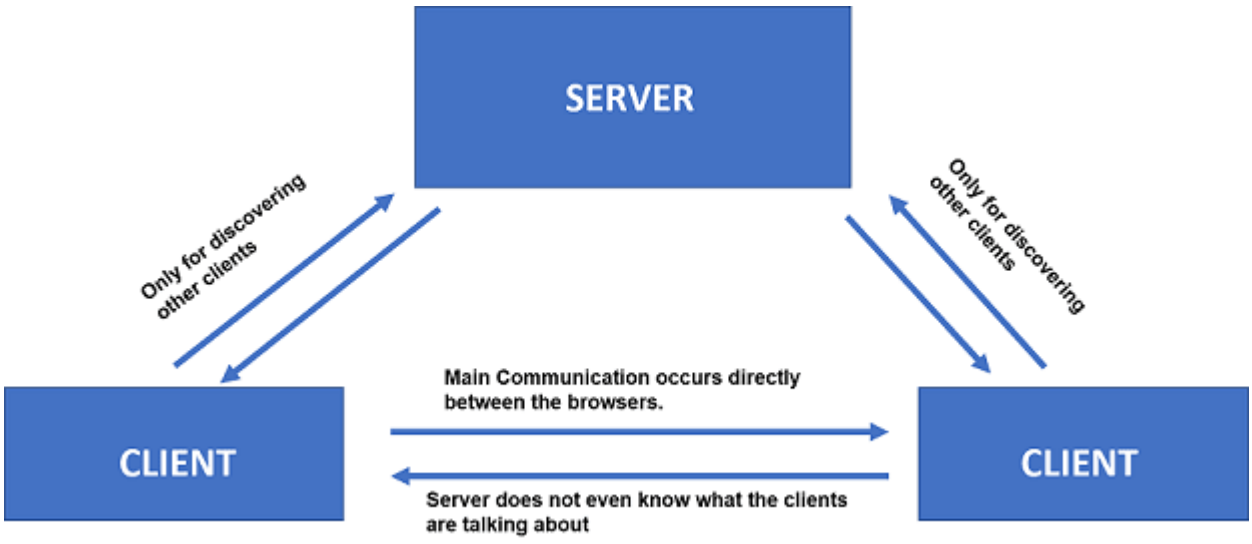


Figure 48: Socket Communication Diagram

To start our video call implementation, we used a couple of libraries to manage our feature:

1- Flutter WebRTC.
   - WebRTC plugin for Flutter Mobile/Desktop/Web.
2- Socket IO Client.
   - To connect the Flutter App with the Socket Server.

3- Flutter Text to Speech.
   - To provide users with sound instructions as well as to give feedback for each event in our app.

4- Firebase and Firestore libraries

(Cloud Firestore, Firebase Core, Firebase Auth, Google Sign in).

- To provide authentication.
- To store info about users in cloud Firestore.
- To provide a Realtime Database which is used in our server (Pyrebase library).

5- State Management libraries (Provider & Flutter Bloc).

- To manage the app and separate its contents into presentation, Business logic and Data layers.
- To provide apps with states that fully cover each event.
- Provide a way of organization.

6- Permission Handler.

- On most operating systems, permissions aren't just granted to apps at install time. Rather, developers must ask the user for permissions while the app is running.

7- Audio libraries (Audio Players & Flutter Ringtone Player).

- As our feature is a video calling feature, we need to provide it with some ringtones and notification sounds.

8- Flask.

- Flask is a web application framework written in python which is used to deal with implementing our socket server

Server while running as shown in figure 49.

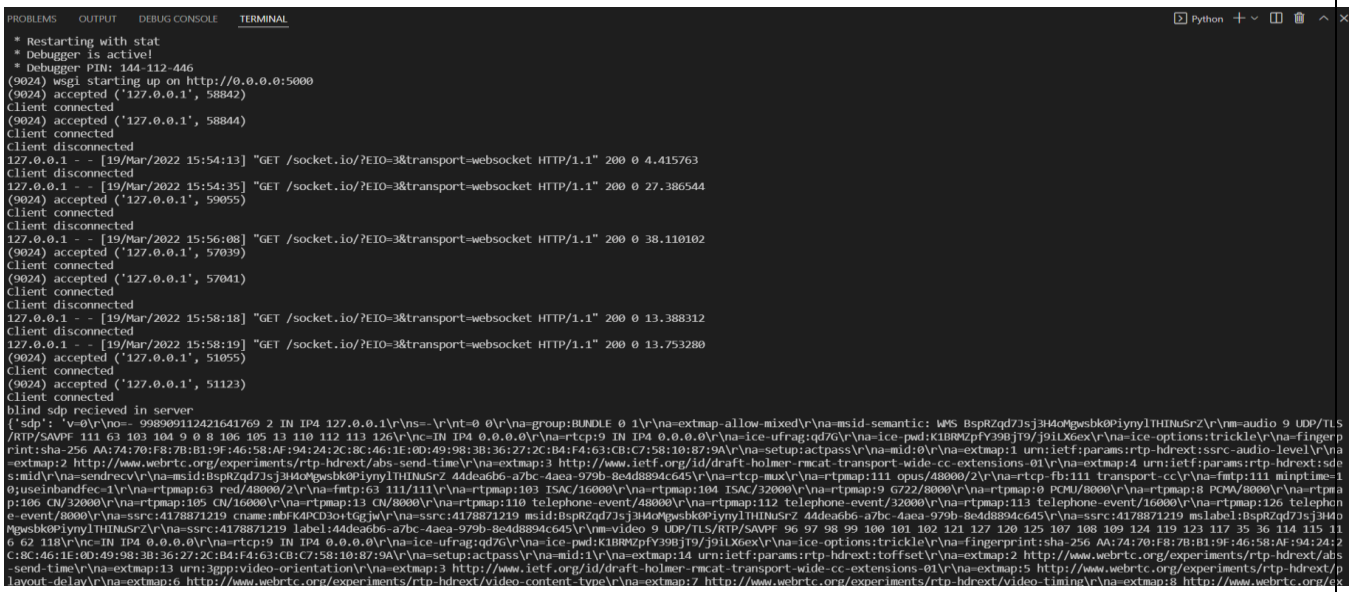


Figure 49: Socket Server Communication Console

Errors facing us in deployment layer:

1- When closing a call on one side the other side is left with a freeze video state.

Solution:

We added a handler on the server to close call on both sides.

2- When multiple blinds are searching for Volunteers, Volunteers are left with multiple calls and when accepting the call falls and makes errors.

Solution:

We solved that by handling the server as no more than one blind call can be received on the other volunteer's side.

3- Blind when calling and one of the volunteers accepts his call the other volunteers are left with a request call from the same blind person.

   Solution:

   This is handled by which if one of the volunteers accepts the call the request on other volunteers' side is canceled.

4- Blind while calling the server is notified only once, it may leave the blind user with no volunteers available at a specific time.

   Solution:

   We increased the percentage of availability of volunteers by notifying the server more than once separated with given periods.

   These and many other errors were handled currently with some methods which can be analyzed and modified in future.

Chapter 7

Chapter 7: Application user interface

1. Blind VS Volunteer Screen as shown in figure50

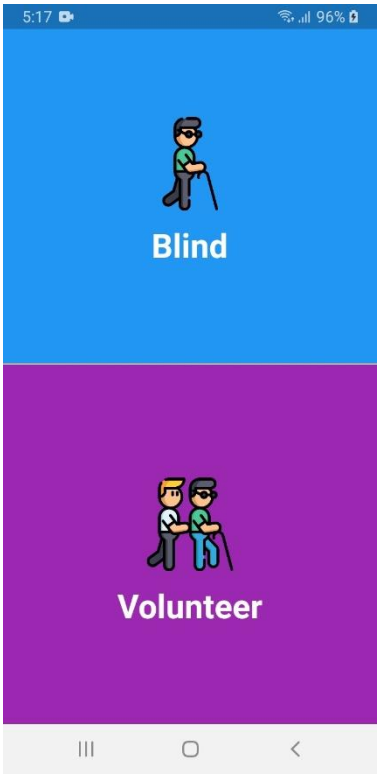


Figure 50: Introduction UI

This is the intro of the application where our users can choose between being a blind user by tapping once on the screen and being a volunteer user by performing a long press on the screen. For any voice guidance in our application, we use text to speech technology.

Voice Assistant:

Welcome to see for me.

For visual aid tap once on the screen.

Else if you are a volunteer long press on the screen.

Actions:

On Tap→ Choose Mode Screen.

On Long Press → App Assistant Screen.

2. Blind Side: Choose Mode Screen as shown in figure51.

Figure 51: Four Mode UI

This is the blind user interface from there our blind user can choose which mode he needs to use.

There are four modes as presented:

App Assistant Mode, Volunteer Assistant Mode, Walk Mode, and Navigation Mode.

Voice Assistant:

Choose your mode.

Tap for app assistant mode.

Double tap for volunteer assistant mode.

Long press for walking mode.

Horizontal drag for Google maps track.

Actions:

On Tap → App Assistant Mode Screen.

On Double Tap → Call Volunteer Mode Screen.

On Long Press → Walk Mode Screen.

On Horizontal Drag → Navigation Mode Screen.

3. **Blind Side: Navigation Mode Screen as shown in figure52.**

   In this mode simply our application calls Google maps with a certain predefined coordinate from here the user can use Google maps via voice assistant and navigate to any desired destination from his current location.

   Application:

   Opens Google Maps with a specific coordinate of Cairo location.

4. **Blind Side: Walk Mode Screen as shown in figure**

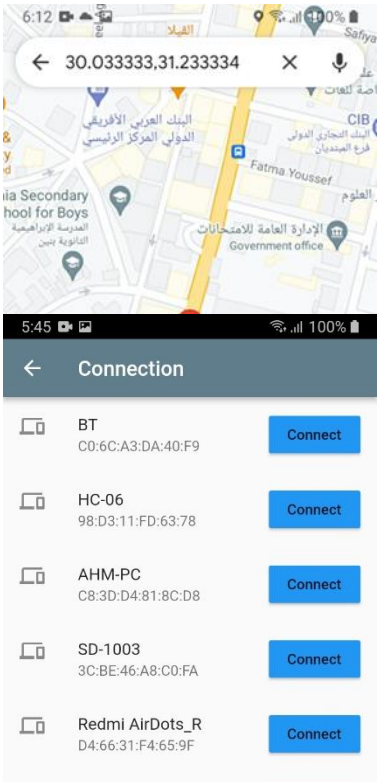


Figure 52: Google Maps UI

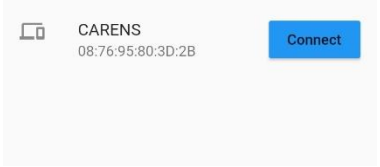


Figure 53: Bluetooth connection UI

53.

In this mode we display all paired Bluetooth devices where our smart shoes gadget (hardware part) is included then the application search for the smart shoes among these devices and connects with it automatically.

Note: smart shoes are defined as HC-06.

Voice Assistant:

Searching for smart shoes.

Application waits for few seconds to search for HC-06 Bluetooth module and connect to it and then pushes Walk Mode Screen 2.

5. **Blind Side: Walk Mode Connected Screen as shown in figure54.**

   When connected to the smart shoes (HC-06) messages are print on the screen displaying the state as well as the distance between the shoes and the nearest discovered obstacle. On detecting a nearby obstacle an alarm sound is activated. Else if the user wants to close this mode all he needs to do is long press on the screen to fully close the application.

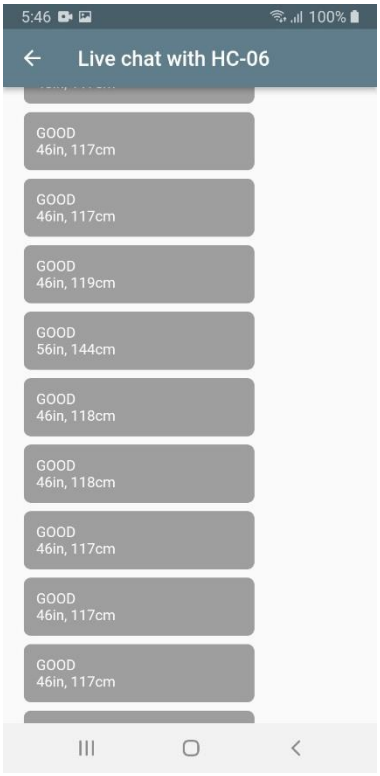


Figure 54: Walk Mode UI 1

States as represented in figure 55.

1- **GOOD** which means that the track is safe. (In our case distance must me greater than 5 inches to be safe).

2- **DANGER** which means that the track contains obstacle and should be rerouted.

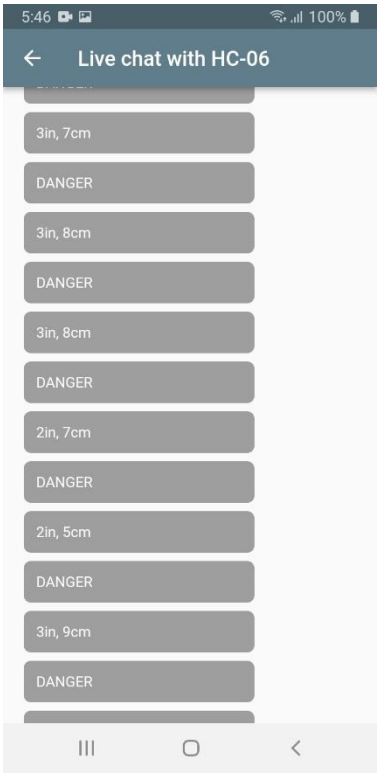


Figure 55: Walk Mode UI 2

Voice Assistant:

Connected to smart shoes. If an obstacle is detected, you will hear the alarm. Therefore, you need to change your direction while walking. If you don't hear the alarm, then your track is safe. Enjoy your walk.

Long press if you want to exit app.

Alarm Sound:

Activated on detecting '**DANGER**' in messages.

Deactivated on detecting '**GOOD**' in messages.

Actions:

On Long press → closes the application.

On Error:

If the application is connected to another device by any mistake the application declares that it is connected to another device and asks the user to either to long press on the screen to close the app or use it for different purpose.

6. **Blind Side: App Assistant Mode Screen as represented in figure 56.**

   In this mode the user can choose between the different features provided by our built AI Models. Simply to do this the user should tap on the screen and say this command like “I want to search for an object” for example
   Then tap again on the screen to stop recording his speech. Then the screen responds with “Opening object detection mode” and pushes the Object Detection Mode Screen.

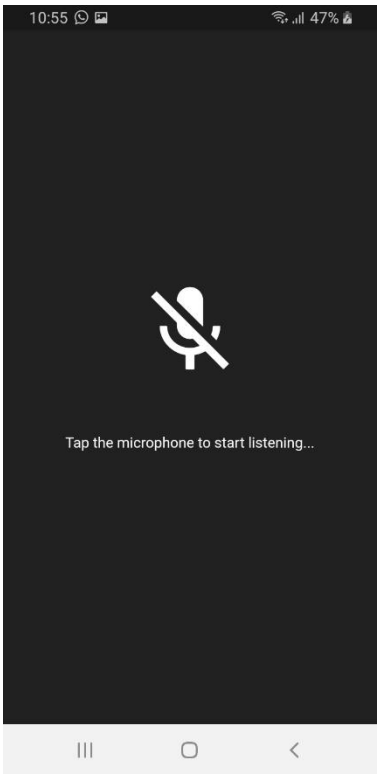


Figure 56: AI Features Assistant UI

   This can be done by the speech to text technology.
   This applies to all features.

   Such features are:

   Face Recognition, Object Detection,
   Cloth Recognition, Text Recognition,
   Banknotes Recognition, Color Recognition, and Indoor Scene Recognition.
   These features are built as AI models that are stored as python files (return to models used topic for more details) so in order to communicate these features with the mobile app we make socket servers to every feature and the reason for that is simple is that every server can work on only one feature using one computer and a block of code that has all the models that works only on one computer. (To improve the performance of servers by dividing them on multiple computers).

Simply those servers contain a socket server as shown in video call part (return to video call for more details about Socket Server). And some simple functions that work with AI models by adding commands as text and images as base64 format images and awaits for response of recognized thing in the form of text then transfer it to the mobile app so that the app responds with the text in the form of speech.

Voice Assistant:

Welcome to see for me assistant mode.

Actions on keyword matching:

As the speech is recorded in the form of text, we scan this text to search its content for a predefined keyword.

If the text contains:

Face, then it opens Face Recognition Screen.

Object, then it opens Object Detection Screen.

Cloth, then it opens Cloth Recognition Screen.

Text, then it opens Text Recognition Screen.

Money, then it opens Banknote Recognition Screen.

Color, then it opens Color Recognition Screen.

Scene – Environment – Surroundings, then it opens Indoor Scene Recognition Screen.

Exit, then it exits the application.

7. **Blind Side: Face Recognition Screen as represented in figure 57.**

   Now that the user had choose the Face Recognition Screen. A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized face.

   (Return to Face Recognition Feature to know the scenarios that the user can use).

Figure 57: Face Identifying UI

   Voice Assistant (for every AI feature screen):

   Welcome to “Face Recognition Mode”, followed by the instructions of what can the user do.

   Actions:

   On Tap → waits for the user to say his command and store it in the form of speech to text.

   Then this recorded text is scanned to get two attributes which are the command keyword and the name of the person in the picture.

   On Double Tap → Checks if the command is taken first then captures the image with focusing the camera and adding flashlight to illuminate the scene before capturing the image and to confirm that an image is captured a camera sound is played after tapping.

   After that we have three attributes which are the captured image, the command keyword, and the person’s name.

   It sends these three attributes to the model and then a feedback response is given in the form text to display to the user in the form of text to speech.

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:

On Tapping user says “Store Ahmed Hisham” then double tap to capture his photo and store it for later use.

Keywords to use are:

Save, Store → to store a person’s image.

Response:

- Saved successfully.

-Try to take a clear picture.

Remove, Delete → to delete a person’s image.

Response:

- Removed successfully.
- Does not exist.

Identify, Verify → to verify if this person is the one in front of the camera.

Response:

- “Person’s name” is here.
- Sorry I could not find the person.
- You did not save “Person’s name”.

8. **Blind Side: Object Detection Screen as shown in figure 58.**

   Else if the user chose the Object Detection Mode Screen. A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized object.

   (Return to Object Detection Feature for further details). Voice Assistant is included as previous screen.

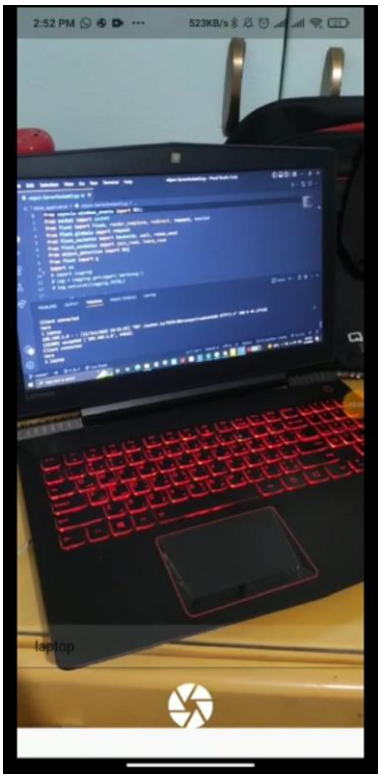


Figure 58: Looking For An Object UI

Actions:

On Tap → waits for the user to specify the object in need to search for. Input of the user is performed and stored in the form of speech to text. Then this recorded text is scanned to get name of the object.

On Double Tap → Checks if the object's name is taken first then captures the image with focusing the camera and adding flashlight to illuminate the scene before capturing the image and to confirm that an image is captured a camera sound is played after tapping.

After that we have two attributes which are the captured image and the object's name.

The application then sends these two attributes to the model and after that a feedback response is given in the form text to display to the user in the form of speech (text to speech).

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:

On tapping user says “Laptop” then double tap to capture the object’s image and then display a response of the “name of the object” found. (In our case “Laptop found”). If the object is not found in the image the application’s response is “not found”. Response is always in the form of text to speech.

9. **Blind Side: Cloth Recognition Screen as represented in figure 59.**

   Else if the user chose the Cloth Recognition Mode Screen. A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized cloth.

   (Return to Cloth Recognition Feature for further details). Voice Assistant is included as previous screen.

Figure 59: Cloth recognition UI

   Actions:

   On Double Tap → Captures the image with focusing the camera and adding flashlight to illuminate the scene before capturing the image and to confirm that an image is captured a camera sound is played after tapping.

   After that we have only one attribute which is the captured image.

   The application then sends the attribute to the model and after that a feedback response is given in the form text to display to the user in the form of speech (text to speech).

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:

On double tap the application captures the image containing cloth and then display a response of the “name of the cloth”. (In our case “Backpack found”). If no cloth found in the image the application’s response is “Cloth not found”. Response is always in the form of text to speech.

10. **Blind Side: Text Recognition Screen as shown in figure 60.**

    Else if the user chose the Text Recognition Mode Screen (that depends on Optical Character Recognition technology). A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized label, document, article, etc.

    (Return to Text Recognition Feature for further details). Voice Assistant is included as previous screen.

Figure 60: Text Recognition UI

Actions:

On Double Tap → Captures the image

with focusing the camera and adding flashlight

to illuminate the scene before capturing the image and to confirm that an image is captured a camera sound is played after tapping.
After that we have only one attribute which is the captured image.
The application then sends the attribute to the model and after that a feedback response is given in the form text to display to the user in the form of speech (text to speech).

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:
On double tap the application captures the image containing text information and then display a response of the “text content”. (In our case “b o b a I Sunscreen cream Rich Texture …”). If no text found in the image the application’s response is “Text not found”. Response is always in the form of text to speech.

11. **Blind Side: Banknotes Recognition Screen as represented in figure 61.**

Else if the user chose the Banknotes Recognition Mode Screen. A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized money paper. (Return to Banknotes Recognition Feature for further details). Voice Assistant is included as previous screen.

Figure 61: Money Recognition UI

Actions:

On Double Tap → Captures the image

with focusing the camera and adding flashlight

to illuminate the scene before capturing the image and to confirm that an image is captured a camera sound is played after tapping.

After that we have only one attribute which is the captured image.

The application then sends the attribute to the model and after that a feedback response is given in the form text to display to the user in the form of speech (text to speech).

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:

On double tap the application captures the image containing paper money and then display a response of the “amount of money”. (In our case “50 EGP”). If no paper money found in the image the application’s response is “Money not found”. Response is always in the form of text to speech.

12. Blind Side: Color Recognition Screen as shown in figure 62.

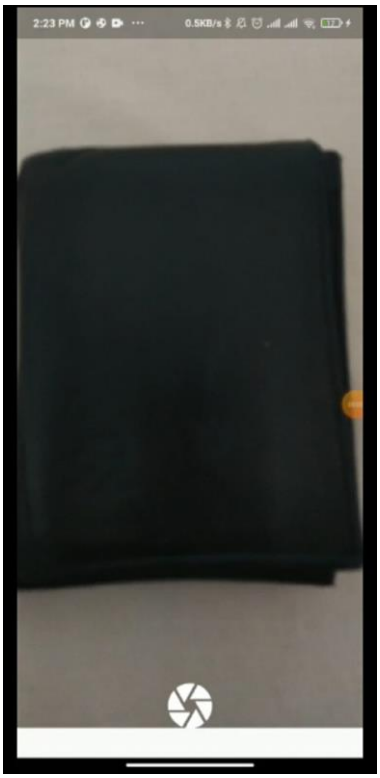


Figure 62: Color detection UI

Else if the user chose the Color Recognition Mode Screen. A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized color.

(Return to Banknotes Recognition Feature for further details). Voice Assistant is included as previous screen.

Actions:

On Double Tap → Captures the image
with focusing the camera and adding flashlight
to illuminate the scene before capturing the
image and to confirm that an image is captured
a camera sound is played after tapping. After that we have only one attribute which is the captured image.

The application then sends the attribute to the model and after that a feedback response is given in the form text to display to the user in the form of speech (text to speech).

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:

On double tap the application captures the image containing colors and then display a response of the “most visible color in the image”. (In our case “black”).

Response is always in the form of text to speech.

13. **Blind Side: Indoor Scene Recognition Screen as shown in figure 63.**

Else if the user chose the Color Recognition Mode Screen. A camera widget is opened to capture an image and send it to the AI model via server connection techniques and receive the response of the recognized indoor scene.

(Return to Indoor Scene Recognition Feature for further details). Voice Assistant is included as previous screen.

Figure 63: Indoor Scene Recognizer UI

Actions:

On Double Tap → Captures the image with focusing the camera and adding flashlight to illuminate the scene before capturing the image and to confirm that an image is captured a camera sound is played after tapping. After that we have only one attribute which is the captured image.

The application then sends the attribute to the model and after that a feedback response is given in the form text to display to the user in the form of speech (text to speech).

On Long Press → returns to the main features menu (App Assistant Mode) screen.

For example:

On double tap the application captures the image containing indoor scene and then display a response of the “indoor scene”. (In our case “Library”). If the image doesn’t include an indoor scene or isn’t distinguishable, the application’s response is “Indoor Scene not found”. Response is always in the form of text to speech.

14. Image Formatting

Why do we use Base64?

Base64 encoded images can be embedded using img tags or CSS, speeding up load times for smaller images by preventing additional HTTP requests. This can be done to build single-file mockups / demo pages for your clients, HTML email signatures that will not trigger the nasty "show images" warning in email clients, etc.

15. Blind Side: Entering Video Call as described in figure 64.

As we defined before our application have four modes which are:

- Navigation Mode
- Walk Mode
- App Assistant Mode

which are discussed in the previous screens.

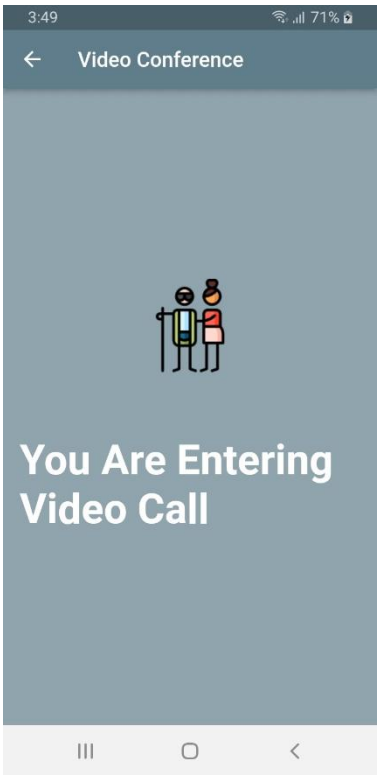


Figure 64: Blind Side User Interface Entering Call

Now we will focus on the fourth and last mode
which is getting in a video call with a volunteer. (Return to Video Call Feature for more details).

As we said before that in order to link the models with the actual mobile app, we need socket servers. Our case here is similar where we need to connect volunteer mobile app side with blind mobile app side so to connect
these two sides we need a socket server to send and receive video data streams between two sides and then communication is achieved.

Voice Assistant:
Entering video call double tab to enter.

Actions:
On Double Tap → Blind user side initializes socket connection, pushes the Blind Offer Screen, and initializes the local video.

16. Blind Side: Offer Screen as shown in figure 65

Here the blind side local video renderer is displayed with a loading icon.

All the blind need to do is long press on the screen to enter the next screen which is the ringing screen and send his peer descriptive side data.

Voice Assistant:

Now you are ready long press to start call.

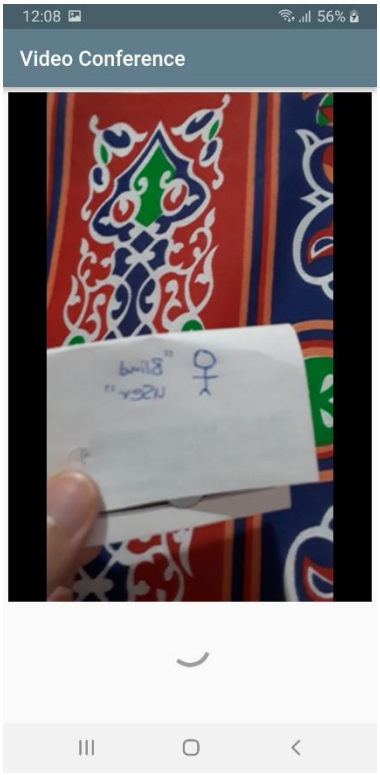


Figure 65: Blind Side User Interface Entering Call 2

17. Blind Side: Ringing Screen as represented in figure 66.

Here the application waits for the server to send the volunteer peer descriptive data side. To configure the remote video stream renderer. A ringtone is displayed for the user to know that he is waiting inside the call.

The application repeats the request every minute. Application waits a few minutes for a volunteer response and if no volunteer is found the application will return to Offer Screen saying that no volunteer is available at the moment.

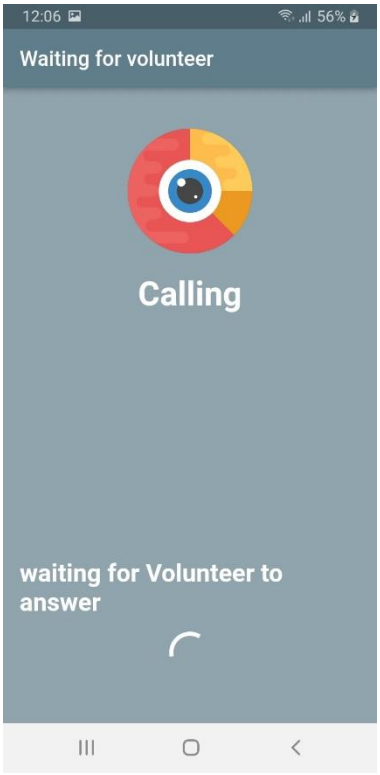


Figure 66: Blind Side User Interface Ringing State

On receiving volunteer peer descriptive data, the blind side sets his remote video renderer and pushes the In Call Screen.

18. Blind Side: In Call Screen as shown in figure 67.

Here the blind enters the video call with the volunteer as their video streams are now connected the back camera of the blind user is used and on the other side the volunteer camera.

Voice Assistant: You are connected with your volunteer assistant.

Actions:

On Tap → blind user mic is toggled (enabled/disabled).

On Double Tap → blind user headset as well as mic are both toggled (enabled/disabled).

On Long Press → blind user closes call going to the next screen which is the Leaving Call Screen.

On Horizontal Drag End → blind user video is toggled (enabled/disabled).

On Vertical Drag End → blind user switches camera (front and back camera).

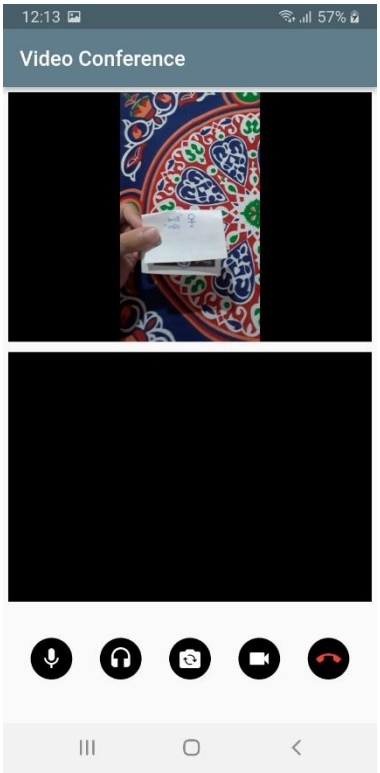
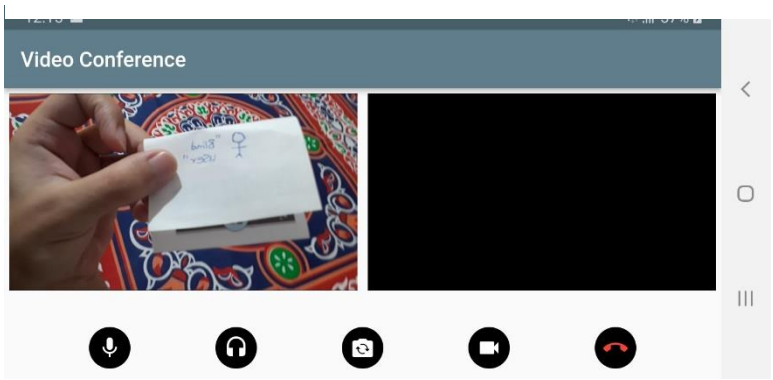


Figure 67: Blind Side In Call User Interface



19. Blind Side: Leaving Call Screen as shown in figure 69.

Here the application is ready to dispose all the resources that is used in the web RTC video call. Leaving the call and disconnecting with the volunteer side.

Then the blind app side is closed.

Figure 68: Blind Side Leaving Call User Interface

Voice Assistant:

Leaving Video Call.

Actions:

On Double Tap → the blind user side is disconnected, and the application is closed.

For more details about Video Call Blind Side return to Video Call Feature.

20. Volunteer Side: Entering Video Call as represented in figure 69

As we defined before our application have two users:

- Blind user.
- Volunteer user.

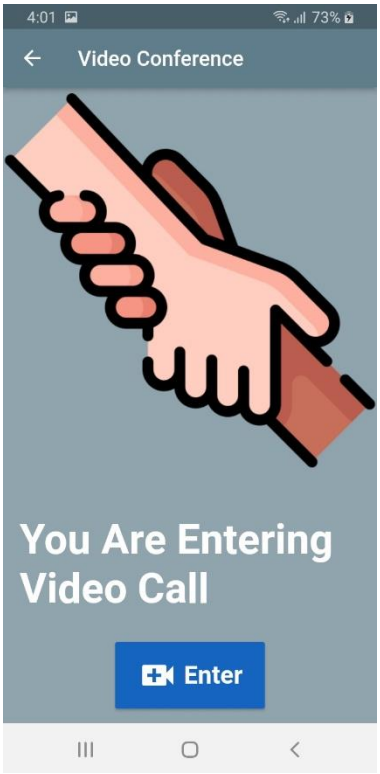


Figure 69: Volunteer Side User Interface Entering Call

Since we covered the blind use of the application now is we will discuss the volunteer interface and his side of the application. (Return to Video Call Feature for more details).

Similar to blind side volunteer side communicates with the socket server in order to join a blind user in a video call by sending his peer descriptive data side.

Actions:
On Button Click → Volunteer user side initializes socket connection, pushes the Volunteer waiting Screen, and initializes the local video.

21. Volunteer Side: Waiting for Blind user Screen as shown in figure 70.

Here the application volunteer side waits for the server response to connect the volunteer to blind user side.
The local video is displayed with loading content.
If the blind user request received the Ringing Screen will be pushed.

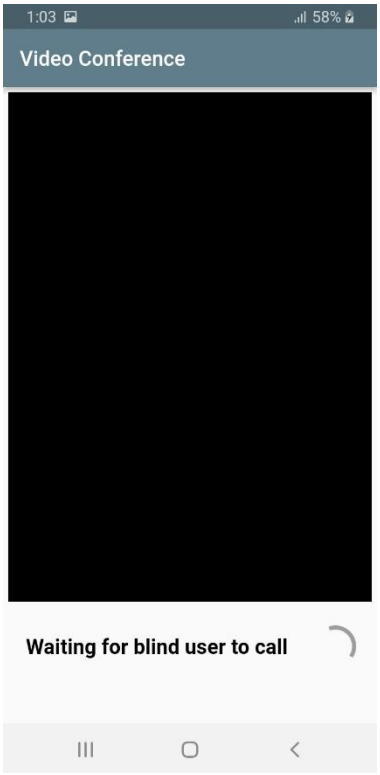


Figure 70: Volunteer Side User Interface Entering Call 2

22. Volunteer Side: Receiving Call Screen as shown in figure 71.

Figure 71: Volunteer Side User Interface Ringing State

On receiving a blind side request to connect to the volunteer side, the volunteer side holds peer descriptive data of the blind side.
Also a ringtone is displayed to alert the volunteer side of the changed state.

Actions:

Green Button Click → sets the blind side peer descriptive data and pushes the In Call Screen.
Red Button Click → returns to the Volunteer Waiting Screen.

23. Volunteer Side: Leaving Call Screen shown in figure 72.

Figure 72: Volunteer Side User Interface Leaving Call

Here the application is ready to dispose all the resources that is used in the web RTC video call.
Leaving the call and disconnecting with the blind side.
Then the volunteer app side is closed.

Actions:

Exit Button Click → the volunteer user side is disconnected, and the application is closed.

24. Volunteer Side: In Call Screen as shown in figure 73.

Here the volunteer enters the video call with the blind user as their video streams are now connected. The volunteer camera is used and on the other side the back camera of the blind user is used.

Actions:

Mic Button → volunteer user mic is toggled (enabled/disabled).

Headset Button → volunteer user headset as well as mic are both toggled (enabled/disabled).

Decline Button → volunteer user closes call going to the next screen which is the Leaving Call Screen.

Video Button → volunteer user video is toggled (enabled/disabled).

Camera Button → volunteer user switches camera (front and back camera).

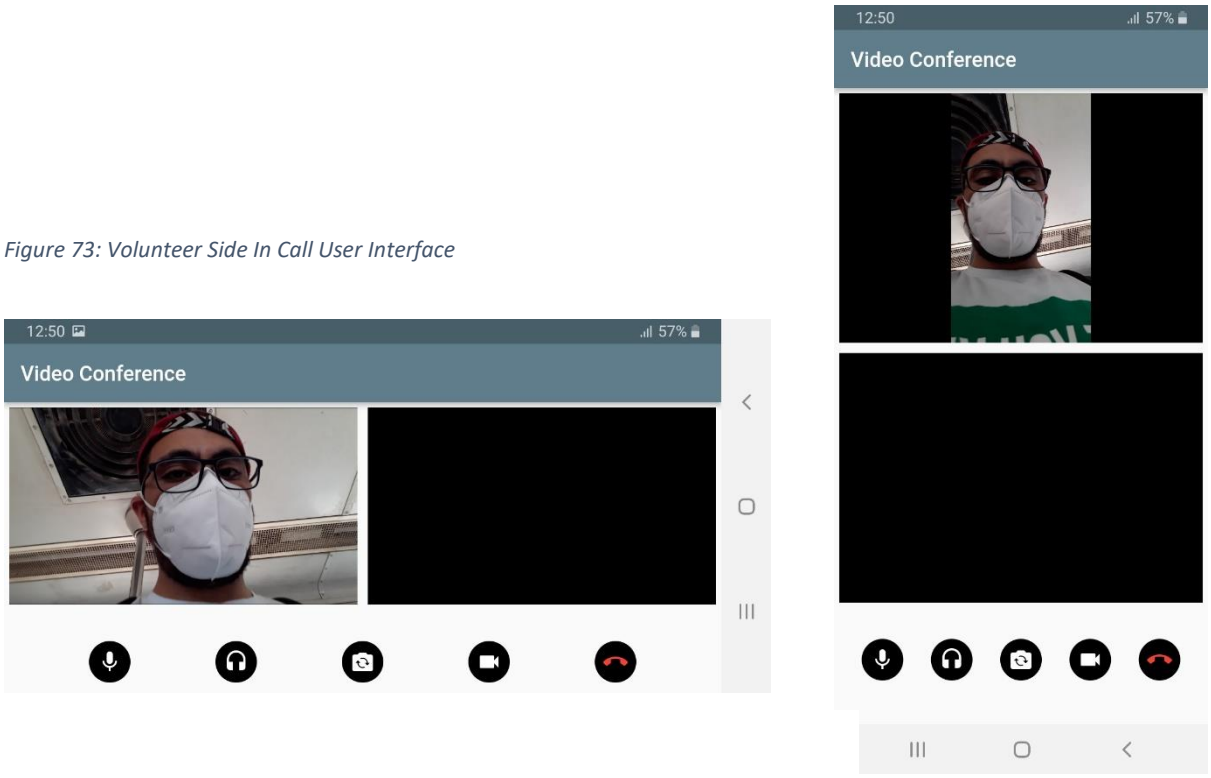


Figure 73: Volunteer Side In Call User Interface

Chapter 8

Chapter 8: Hardware

8.1 Object detector:

This part is used for sensing the objects when any object is close to the system the inches are less than 5 the alarm will be ON.

Main components:

Bread board, LEDs, Resistor -220 ohms, Male to male jumper wire, Bluetooth module, arduino uno, arduino ultrasonic module (HC-SR04), and Supply 9V Battery.

8.2 Arduino uno:

Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

-Figure 74 represents the Arduino Uno.

Figure 74: Arduino Uno

8.3 Arduino ultrasonic module (HC-SR04):

This is the most popular sensor for measuring distance and making obstacles avoiding robots with Arduino. The HC-SR04 is an affordable and easy to use distance measuring sensor which has a range from 2cm to 400cm (about an inch to 13 feet). The sensor is composed of two ultrasonic transducers. One is the transmitter which outputs ultrasonic sound pulses and the other is the receiver which listens for reflected waves.

-Figure 75 represents Ultrasonic HC-SR04.

Figure 75: Ultrasonic HC-SR04 Sensor

8.4 Bluetooth module:

Bluetooth Module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. Its communication is via serial communication which makes an easy way to interface with a controller or PC.

-Figure 76 represents the Bluetooth module HC-06.

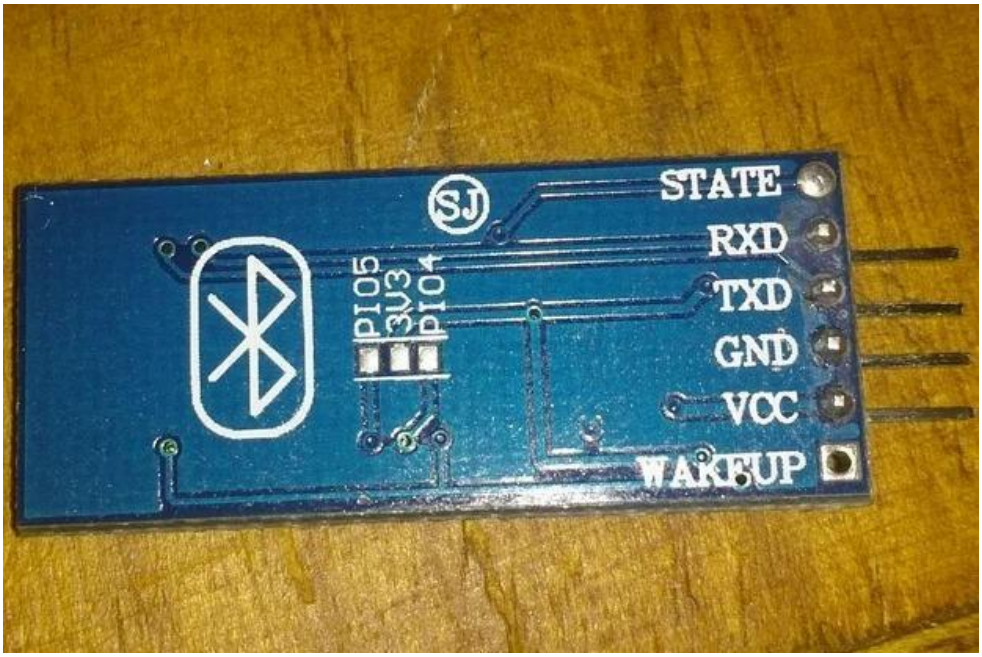


Figure 76: Bluetooth HC-06 Module

8.5 Connections:

-Figure 77 represent the connections of the component as discussed before.

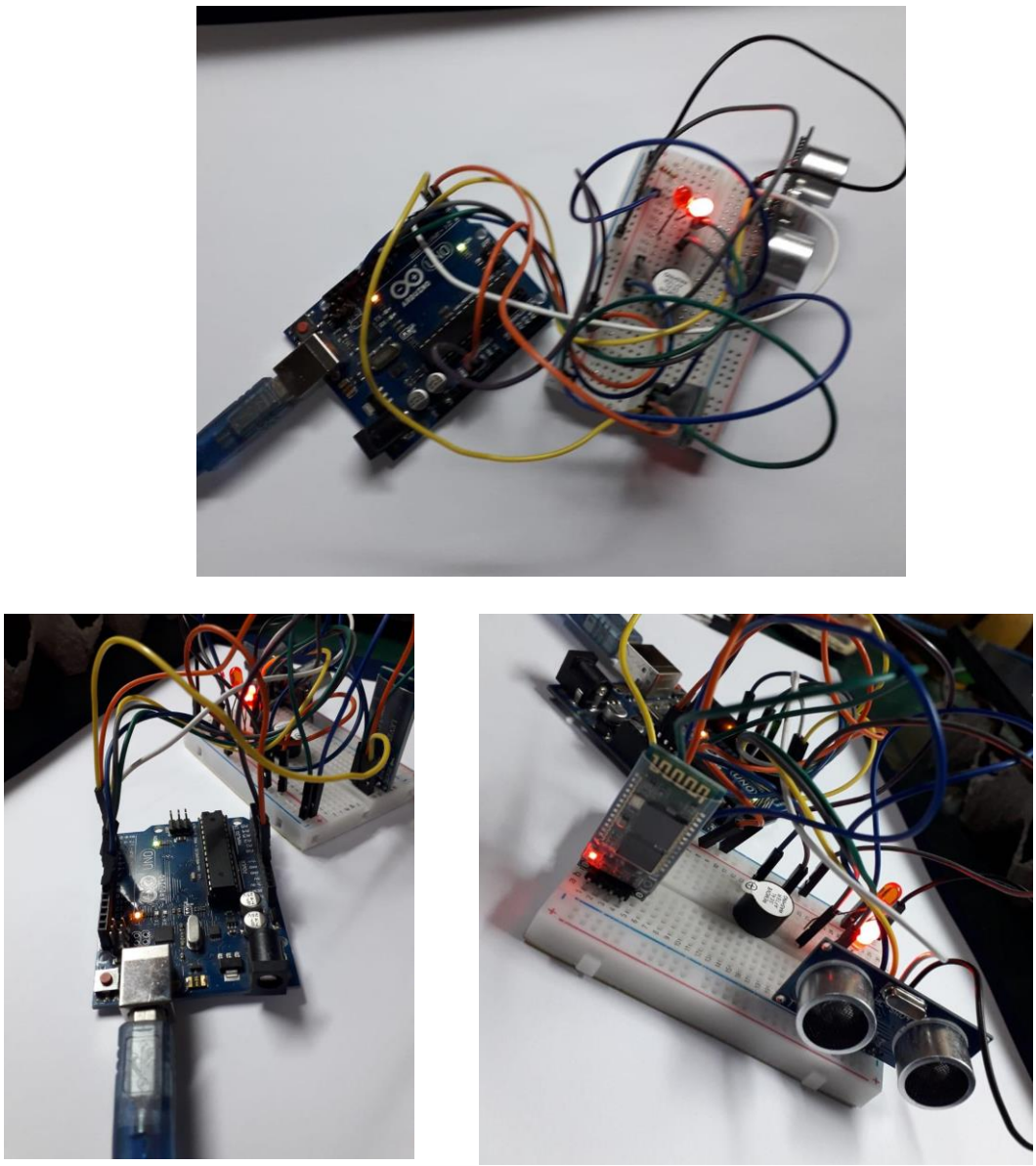


Figure 77: Smart Shoes Circuit Implementation

References:

1)) Ali, Maghfirah & Tang, Tong Boon. (2016). Smart Glasses for the Visually Impaired People. 9759. 579-582. 10.1007/978-3-319-41267-2_82.

2) Joshi, Vijay & A, Visu & Sivakumar, Mohan Raj & T, Madhan & G, Kalaiselvi. (2011). PENPAL - Electronic Pen Aiding Visually Impaired in Reading and Visualizing Textual Contents. 2012 IEEE Fourth International Conference on Technology for Education. 171-176. 10.1109/T4E.2011.34.

3) Edan, Naktal & Al-Sherbaz, Ali & Turner, Scott. (2018). Design and Implement a Hybrid WebRTC Signalling Mechanism for Unidirectional & Bi-directional Video Conferencing. International Journal of Electrical and Computer Engineering. 8. 390-399. 10.11591/ijece.v8i1.pp390-399.

3) Ijrece, & Vol, & Kumar, Ashish. (2018). Smart Blind stick for Visually Impaired People.

4) Awad, Milios & Haddad, Jad & Khneisser, Edgar & Mahmoud, Tarek & Yaacoub, E. & Malli, Mohammad. (2018). Intelligent eye: A mobile application for assisting blind people. 1-6. 10.1109/MENACOMM.2018.8371005.

5) Kaur, Parminder & Ganore, Mayuri & Doiphode, Rucha & Garud, Ashwini & Ghuge, Tejaswini. (2017). Be My Eyes : Android App for visually impaired people. 10.13140/RG.2.2.12307.48164

6) M. S. R. Tanveer, M. M. A. Hashem and M. K. Hossain, "Android assistant EyeMate for blind and blind tracker," 2015 18th International Conference on Computer and Information Technology (ICCIT), 2015, pp. 266-271, doi: 10.1109/ICCITechn.2015.7488080.

7) M. Liwicki, E. Indermuhle and H. Bunke, "On-Line Handwritten Text Line Detection Using Dynamic Programming," Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 2007, pp. 447-451, doi: 10.1109/ICDAR.2007.4378749.

8) Zhu Z, Gao Y and Gu S. 2021. Tennis Ball Collection Robot Based on MobileNet-SSD 2021 11th International Conference on Intelligent Control and Information Processing (ICICIP). 10.1109/ICICIP53388.2021.9642172. 978-1-6654-2515-5, (300-307).

9) C. P. Papageorgiou, M. Oren and T. Poggio, "A general framework for object detection," Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271), 1998, pp. 555-562, doi: 10.1109/ICCV.1998.710772.

10) Handwriting Recognition using Artificial Intelligence Neural Network and Image Processing Sara Aqab1 , Muhammad Usman Tariq2 Abu Dhabi School of Management Abu Dhabi, UAE.

11) V. M. Safarzadeh and P. Jafarzadeh, "Offline Persian Handwriting Recognition with CNN and RNN-CTC," 2020 25th International Computer Conference, Computer Society of Iran (CSICC), 2020, pp. 1-10, doi: 10.1109/CSICC49403.2020.9050073.

12) Scene recognition **:**A. Quattoni and A. Torralba, "Recognizing indoor scenes," 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 413-420, doi: 10.1109/CVPR.2009.5206537.

13) Kaur, Parminder & Ganore, Mayuri & Doiphode, Rucha & Garud, Ashwini & Ghuge, Tejaswini. (2017). Be My Eyes : Android App for visually impaired people. 10.13140/RG.2.2.12307.48164.

14) Majid, Hairudin & A Samah, Azurah & Mi Yusuf, Lizawati & Nasien, Dewi & Cheah, T.L.. (2016). P2P audio and video calling application using WebRTC. 11. 1766-1770.

15) Kushtrim Pacaj & Kujtim Hyseni & Donika Sfishta,(2020),Peer to Peer Audio and Video Communication using WebRTC,EasyChair Preprint no. 4304.

16) N. Pathania, R. Singh, isha and A. Malik, "Comparative Study of Audio and Video Chat Application Over the Internet," 2018 International Conference on Intelligent Circuits and Systems (ICICS), 2018, pp. 251-257, doi: 10.1109/ICICS.2018.00059.