



# Infant Drowning Prevention and Alert System using Video Vision Transformer

*Mrs. R. Priyadharshini M. E.* <sup>\*1</sup>, *Mr. Ajay Kishore. E* <sup>\*2</sup>, *Mr. Bharath. P* <sup>\*3</sup>, *Mr. Gokulakrishnan. M* <sup>\*4</sup> *Mr. Sanjay Kumar.A* <sup>\*5</sup>

<sup>\*1</sup>Assistant Professor of computer science and engineering, MRK Institute of Technology, Kattumanarkoil, Tamil Nadu

<sup>\*2</sup>UG Student of Computer science and engineering, MRK Institute of Technology, Kattumanarkoil, Tamil Nadu

<sup>\*3</sup>UG Student of Computer science and engineering, MRK Institute of Technology, Kattumanarkoil, Tamil Nadu

<sup>\*4</sup>UG Student of Computer science and engineering, MRK Institute of Technology, Kattumanarkoil, Tamil Nadu

<sup>\*5</sup>UG Student of Computer science and engineering, MRK Institute of Technology, Kattumanarkoil, Tamil Nadu

## ABSTRACT

Drowning deaths represent the third leading cause of accidental deaths worldwide. Drowning accidents in indoor swimming pools are growing in recent years, especially for children. This is because traditional techniques for the supervision and care of people, especially children, in large pools are inefficient or, in some cases, non-existent. Nowadays, this problem has become a topic of interest for several researchers who seek to propose different methods of drowning detection. This project seeks to propose the process to be followed to develop a drowning detection system in swimming pools using Video Vision Transformer. This project proposes a novel embedding scheme and a number of Transformer variants to model video clips. This model extracts spatio-temporal tokens from the input video, which are then encoded by a series of transformer layers. The output tokens of the pre-trained ViViT contain spatio-temporal information of drowning scenes. Then transform a query scene and candidate scenes into output token features using the pre-trained ViViT and calculate the similarity between the tokens with cosine similarity. The proposed network is designed to be lightweight based on the Temporal Transformer and Feature Pyramid Networks to detect drowning infants underwater instead of large networks. Therefore, the creation of automated systems that are able to provide with accurate alarms when certain events take place is of paramount importance, as this can heavily reduce swimming pool drowning accidents and improve the efficiency of the system. The experimental results show that the proposed network can ensure relatively good accuracy with fast detection speed.

Keywords: Infant, Drowning, Prevention, Alert System, Video, Vision Transformer

## 1. INTRODUCTION

### 1.1 OVERVIEW

One of the main causes of death for kids and teenagers is drowning. Most drowning deaths involving little children occur in residential pools or hot tubs. Drowning occurs more frequently in rivers, lakes, and oceans among teenagers. It generally happens silently and quickly. Suffocation caused by immersion of the mouth and nose in liquid is known as drowning. A respiratory impairment brought on by submersion or immersion in water is called drowning.

A youngster is most likely to drown when they are left alone or experience a momentary distraction (World Health Organization, 2023). The majority of fatal drowning incidents happen when the victim is alone themselves or when there are witnesses but are unable to save them. The World Health Organization (WHO) and other medical organizations describe drowning as respiratory impairment, or the inability to breathe as a result of submersion. The WHO defines "drowning" as both fatal and nonfatal, even though the term has historically only been used to describe situations that end in death in the sea. Every year, 3,957 Indian newborns lose their lives due to drowning, with children between the ages of one and four being most vulnerable. Actually, second only to vehicle accidents as the primary cause of unintentional mortality among children aged 1 to 14 is drowning. A significant number of drowning deaths and nonfatal injuries occurred among children under the age of fifteen. Between 2018 and 2023, there were an average of 371 drowning deaths and 8,300 hospital admissions due to drowning injuries in children under the age of fifteen. A kid drowning can be extremely tragic and have a lasting impact on the family. Since there may not be enough time to save the child who is drowning, all effort should be focused on prevention.

### 1.2 PROBLEM DEFINATION

According to WISQARS data from the Centers for Disease Control and Prevention (CDC) for the years 2017–2023.

For kids aged 1-4, accidental drowning was the most common cause of death. Drowning was the second most common unintentional death cause for children aged 5 to 14 (after motor vehicle crashes). Every year, about 900 kids and teenagers between the ages of 0 and 19 pass away from accidental drowning. That translates to three drowning deaths on average every day. Children of Indian, American, Alaska Native, and Black descent have a markedly increased risk of drowning. When it came to drowning, boys were more than twice as likely as girls. It takes less than a minute to drown. Parents frequently misjudge the speed at which danger can materialize in the water. Throughout their entire time in or near the water, even young children who are proficient swimmers should be attentively watched. This holds true whether you're at your own backyard pool, the beach, a water park, or a communal pool. A child's risk of drowning in a pool increases from age one to four. Young children typically have drowning accidents near to home. For children aged 1-4, backyards and community pools pose the greatest danger of drowning. The risk of drowning is silent. Drowning scenes in TV shows and movies are frequently followed by a lot of splashing, shrieking, and chaos. As it happens, drowning is frequently quick and quiet. Because of this, it's critical to constantly monitor the pool whenever kids are allowed to enter the water. The places where drowning is most likely to occur vary in age. Two-thirds of drowning deaths among newborns under one year old happen in bathtubs. Children between the ages of 5 and 14 drown in natural water around 40% of the time, and in swimming pools about 30% of the time. When it comes to individuals aged 15 and above, natural waters such as lakes, rivers, or oceans account for more than half of both fatal and nonfatal drownings (CDC, 2023). Currently, child supervision and management are ineffective. The majority of drowning incidents involve kids playing or swimming unsupervised or in locations where swimming is prohibited. Furthermore, the living conditions remain unsafe. Many lakes and rivers lack barriers, and many deep and deadly ponds lack warning signs. Many kids even end up drowning after falling into their own families' water tanks. Additionally, there is a lack of facilities, teachers, and equipment for teaching kids safety and swimming abilities. As kids get older, the risks of drowning and the ways to prevent it shift, but numerous levels of prevention are necessary for everyone. Government regulations and law enforcement on environmental modification, together with parental unwavering focus and monitoring around children while they are near bodies of water, can significantly contribute to preventing the untimely loss of a loved one.

---

## 2. AIM AND OBJECTIVE

The aim of the project is to develop a technology-driven solution that leverages advanced computer vision techniques, specifically Video Vision Transformers (VVTs), to enhance the safety of infants around water bodies by detecting potential drowning incidents and alerting caregivers in real-time. The primary goal is to minimize the risk of infant drowning accidents and provide caregivers with timely information to take preventive actions.

### Other objectives of the project may include:

- ✓ To Develop VVT Model: Create and train a Video Vision Transformer model for analyzing sequential video frames to detect infants and potential drowning scenarios.
- ✓ To Build Annotated Dataset: Curate a dataset of videos depicting infants near water, annotated with safe and risky behaviors for model training.
- ✓ To Enable Real-Time Monitoring: Establish a system with strategically placed cameras to capture continuous video streams near water bodies.
- ✓ To Recognize Infant Behavior: Implement object recognition within the VVT model to differentiate safe and hazardous behaviors.
- ✓ To Detect Anomalies: Develop an anomaly detection mechanism to identify distress or dangerous activities involving infants.
- ✓ To Generate Alerts: Create a system that generates immediate alerts to caregivers and adults upon detecting potential drowning situations.
- ✓ To Ensure Privacy: Address privacy concerns by implementing secure data handling and communication protocols.
- ✓ To Adapt and Improve: Design mechanisms for the system to learn from real-world scenarios and enhance accuracy over time.
- ✓ To Provide User Interface: Develop an intuitive interface for caregivers to monitor alerts and system status.
- ✓ To Test in Real Scenarios: Rigorously validate the system's effectiveness in real-world scenarios to ensure accurate detection and alert generation.

---

## 3. SYSTEM REQUIREMENT AND SPECIFICATON

### 3.1. HARDWARE REQUIREMENTS

- **Server Infrastructure:**
  - Processors: Intel® Core™ i5 processor, 8 GB of DRAM
  - Disk space: 320 GB
- **Camera Hardware:**
  - Web Camera.
- **User Devices:**

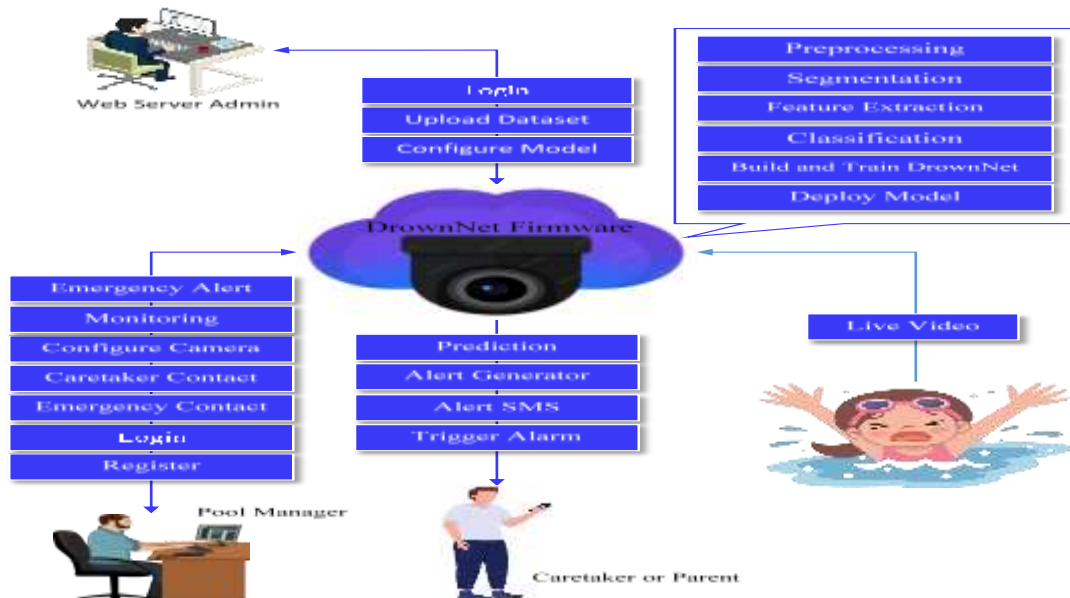
- PCs or laptops for administrative tasks.
- Mobile devices (smartphones, tablets) for caregivers to receive alerts.

### 3.2. SOFTWARE REQUIREMENTS

- **Web Application Development:**
  - **Python:** Primary programming language for backend development.
  - **Flask:** Lightweight web application framework for Python.
  - **HTML/CSS/JavaScript:** Frontend development languages for creating the user interface.
  - **Bootstrap:** Frontend framework for designing responsive and mobile-friendly interfaces.
- **Database Management:**
  - **MySQL:** Relational database management system for storing system data.
- **Deep Learning and Computer Vision:**
  - **TensorFlow or PyTorch:** Deep learning frameworks for implementing the Video Vision Transformer (VVT) and Convolutional Neural Network (CNN) in the DrownNet model.
- **Communication and Notification:**
  - **Pay4SMS:** APIs for integrating SMS notification capabilities.
  - **SMTP Server:** Simple Mail Transfer Protocol for sending email notifications.
- **Server and Hosting:**
  - **Windows 11:** Operating system for the server infrastructure.
  - **Apache:** Web servers for hosting the Flask application.

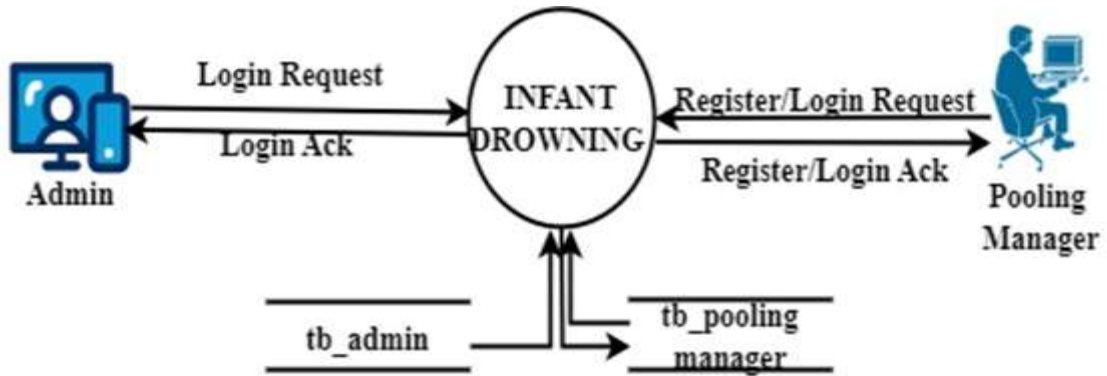
## 4. SYSTEM AND ARCHITECTURE

### 4.1 SYSTEM ARCHITECTURE

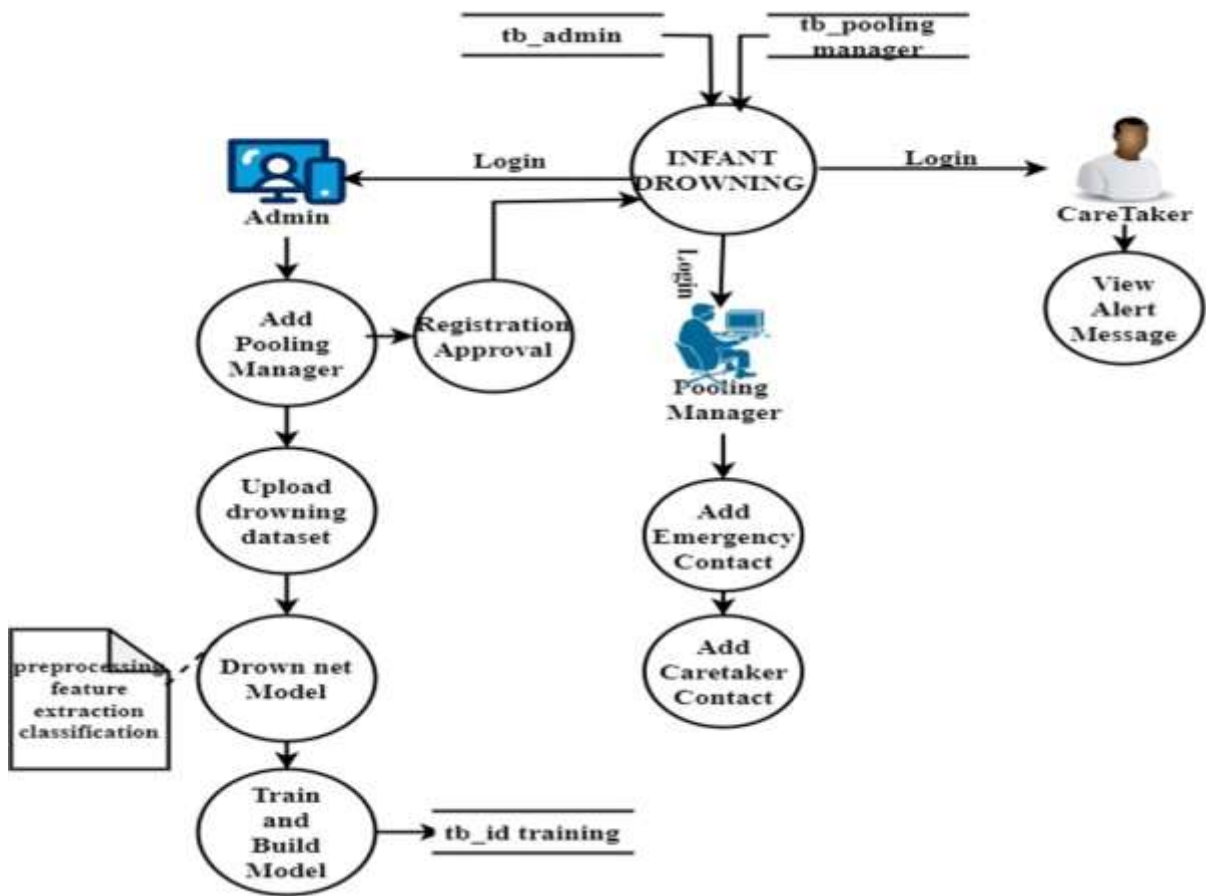


4.2 DATA FLOWDIAGRAM

LEVEL 0



LEVEL 1





3. **Alarm Generation:** The system successfully generated accurate alarms upon detecting potential drowning incidents. By transforming query and candidate scenes into output token features and calculating similarity using cosine similarity, the system effectively identified instances requiring immediate attention, thus enabling timely intervention.

---

## 6. DISCUSSION:

The results obtained from the experimentation underscore the effectiveness and feasibility of the proposed drowning detection system based on Video Vision Transformer. Several key points emerged during the discussion of the project's outcomes:

1. **Accurate Detection:** The high accuracy achieved by the system highlights its potential for reducing drowning accidents in swimming pools. By leveraging advanced deep learning techniques and transformer-based architectures, the system demonstrated the capability to reliably identify drowning incidents with minimal false positives.
2. **Efficiency and Speed:** The efficient design of the network architecture, incorporating lightweight components, contributed to the system's fast detection speed. This ensures rapid response times, crucial for preventing accidents and ensuring the safety of individuals in pool environments.
3. **Potential Impact:** The successful implementation of an automated drowning detection system holds significant implications for enhancing pool safety and reducing drowning-related fatalities. By providing real-time alerts and accurate alarms, the system empowers caregivers and pool operators to respond promptly to emergencies, potentially saving lives.
4. **Future Directions:** Further research and development efforts can focus on enhancing the scalability and adaptability of the system. Integration with advanced monitoring technologies and real-world deployment in diverse pool environments can provide valuable insights for optimizing system performance and addressing practical challenges.

In conclusion, the results and discussion highlight the effectiveness and potential of the proposed drowning detection system using Video Vision Transformer. By leveraging cutting-edge technologies, the system offers a promising solution for improving pool safety and mitigating the risks of drowning accidents.

---

## 7. CONCLUSION

In conclusion, this project stands at the forefront of technological innovation in ensuring the safety of infants around water bodies. By harnessing the capabilities of the Video Vision Transformer (VVT) and the trained DrownNet Model, the system demonstrates remarkable accuracy in the real-time detection of potential drowning risks. Its proactive alerting mechanism serves as a crucial lifeline, delivering immediate notifications to caregivers and facilitating prompt intervention in critical situations. The user interfaces, including the Web Admin, Pool Manager, Pool Camera, and DrownNet Model interfaces, collectively contribute to a comprehensive and user-friendly system. Administrators benefit from secure access and flexible configuration options, while pool managers gain oversight and control over camera setups and emergency contacts. Caregivers receive real-time alerts through multiple channels, ensuring they stay informed and can respond swiftly to any potential threats. The system's adaptability to diverse environments, coupled with the Historical Data and Reporting Module, fosters a culture of continuous improvement. The insights gained from historical data allow for refinements in the model and system configuration, ensuring ongoing efficacy and responsiveness to emerging challenges. In essence, the Infant Drowning Prevention and Alert System not only represents a technological milestone but also a compassionate and practical approach to safeguarding the well-being of infants. Its multifaceted design, proactive features, and commitment to ongoing enhancement position it as a valuable tool in creating safer aquatic environments and preventing tragic incidents of infant drowning.

---

## 8. BIBLIOGRAPHY

1. C Y Wang, A Bochkovskiy and H Y M Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors[C]", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7464-7475, 2023
2. Y. Liu and C. H. Wu, "Drowning incidents and conditions due to hidden flash rips in Lake Michigan", *Sci. Total Environ.*, vol. 827, Jun. 2022.
3. S. Jalalifar, A. Kashizadeh, I. Mahmood, A. Belford, N. Drake, A. Razmjou, et al., "A smart multi-sensor device to detect distress in swimmers", *Sensors*, vol. 22, no. 3, 2022.
4. F. Lei, H. Zhu, F. Tang and X. Wang, "Drowning behavior detection in swimming pool based on deep learning", *Signal Image Video Process.* 2022 166, vol. 16, no. 6, pp. 1683-1690, Jan. 2022.
5. F Wang, Y Ai and W Zhang, "Detection of early dangerous state in deep water of indoor swimming pool based on surveillance video[J]", *Signal Image and Video Processing*, vol. 16, no. 1, pp. 29-37, 2022.
6. K N Bhargavi and G J Suma, "Identifying Drowning Objects in Flood Water and Classifying using Deep Convolution Neural Networks[J]", *i-manager's Journal on Image Processing*, vol. 8, no. 3, pp. 1-14, 2021.

7. S Hasan, J Joy, F Ahsan et al., "A Water Behavior Dataset for an Image-Based Drowning Solution[C]", *2021 IEEE Green Energy and Smart Systems Conference (IGESSC)*, pp. 1-5, 2021.
8. T Peng, J Shen and Y Qiao, "Design of Swimming Pool Drowning Behavior Detection System Based On Improved Mask R-CNN[J]", *Transducer and Microsystem Technologies*, vol. 40, no. 01, pp. 94-97, 2021.
9. Peng Ting, Shen Jinghu and Qiao Yu, "Design of a pool drowning behavior detection system based on improved Mask R-CNN[J]", *Sensors and Microsystems*, vol. 40, no. 01, pp. 94-97, 2021.
10. Wang Yirong, Cai Weicong and Lei Lin, "Epidemiological Characteristics and Research Progress of Prevention Measures for Child Drowning [J]", *Injury Medicine (Electronic Edition)*, vol. 9, no. 01, pp. 61-67, 2020.
11. Carballo-Fazanes and J. J. L. M. Bierens, "The Visible Behaviour of Drowning Persons: A Pilot Observational Study Using Analytic Software and a Nominal Group Technique", *International Journal of Environmental Research and Public Health*, vol. 17, no. 18, pp. 6930-6943, September 2020.
12. Alotaibi, "Automated and Intelligent System for Monitoring Swimming Pool Safety Based on the IoT and Transfer Learning", *Electronics*, vol. 9, no. 12, pp. 2082, December 2020.
13. I. N. Alshbatat, S. Alhameli, S. Almazrouei, S. Alhameli and W. Almarar, "Automated Vision-based Surveillance System to Detect Drowning Incidents in Swimming Pools", in *2020 Advances in Science and Engineering Technology International Conferences (ASET)*, pp. 1-5, 2020.
14. Claesson, S. Schierbeck, J. Hollenberg, S. Forsberg, P. Nordberg, M. Ringh, et al., "The use of drones and a machine-learning model for recognition of simulated drowning victims--- A feasibility study", *Resuscitation*, vol. 156, pp. 196-201, November 2020.
15. Qiao Yu, "Design and implementation of Mask R-CNN based drowning behavior detection system in swimming pools[D]", *Qingdao University*, 2019.
16. Pingping Zhu, Jason Isaacs, Bo Fu and Silvia Ferrari, "Deep learning feature extraction for target recognition and classification in underwater sonar images", *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 2724-2731, 2017.
17. N. Salehi, M. Keyvanara and S. Monadjemmi, "An Automatic Video-based Drowning Detection System for Swimming Pools Using Active Contours", *I.J. Image Graphics and Signal Processing*, vol. 8, no. 8, 2016.
18. Z. Chi, X. Li and F. Lei, "A Novel Camera-Based Drowning Detection Algorithm" in *Advances in Image and Graphics Technologies*, Berlin Heidelberg:Springer, pp. 224-233, 2015.
19. W. Wong, J. Hui, C. Loo and W. Lim, "Off-time Swimming Pool Surveillance Using Thermal Imaging System", *International journal of innovative computing information and control*, vol. 9, no. 3, pp. 366-371, 2013.
20. R. Dubois, D. Thiel and D. James, "Using Image Processing for Biomechanics Measures in Swimming", *Procedia Engineering*, vol. 34, pp. 807-812, 2012.

## BOOK REFERENCES

1. "Deep Learning with Python" by Francois Chollet: This book provides an in-depth introduction to deep learning with Python and covers topics such as neural networks, convolutional neural networks, and natural language processing.
2. "TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning" by Bharath Ramsundar and Reza Bosagh Zadeh: This book provides a comprehensive introduction to TensorFlow, one of the most widely used deep learning frameworks, and covers topics such as neural networks, convolutional neural networks, and recurrent neural networks.
3. "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurelien Geron: This book provides a practical and hands-on approach to machine learning and covers topics such as data preprocessing, classification, regression, and clustering.
4. "Python Machine Learning" by Sebastian Raschka and Vahid Mirjalili: This book provides an introduction to machine learning with Python and covers topics such as supervised learning, unsupervised learning, and deep learning.
5. "MySQL for Python" by Albert Lukaszewski: This book provides an introduction to MySQL and covers topics such as database design, data modeling, and SQL queries.

## WEB LINK REFERENCES

1. TensorFlow documentation: <https://www.tensorflow.org/>
2. Flask documentation: <https://flask.palletsprojects.com/en/2.1.x/>
3. MySQL documentation: <https://dev.mysql.com/doc/>
4. OpenCV documentation: <https://docs.opencv.org/>

5. PyAudio documentation: <https://people.csail.mit.edu/hubert/pyaudio/docs/>
6. Deep Learning for Computer Vision with Python:
7. <https://www.pyimagesearch.com/deep-learning-computer-vision-python-book/>
8. Coursera: Deep Learning Specialization:
9. <https://www.coursera.org/specializations/deep-learning>
10. American Academy of Pediatrics. (2020). Drowning Prevention Campaign Toolkit. Retrieved from <https://www.aap.org/en/news-room/campaigns-and-toolkits/drowning-prevention/>
11. Centers for Disease Control and Prevention. (2022). Drowning Facts. Retrieved from <https://www.cdc.gov/drowning/facts/index.html>
12. Centers for Disease Control and Prevention. (2017-2020). Web-based Injury Statistics Query and Reporting System (WISQARS).
13. World Health Organization. (2023). Drowning. Retrieved. <https://www.who.int/news-room/fact-sheets/detail/drowning>

---

## APPENDIX

### SAMPLE CODING

#### Packages

```
import io
import math
from flask import Flask, render_template, Response, redirect, request, session, abort, url_for
from camera import VideoCamera
import mysql.connector
import hashlib
import datetime
import calendar
import random
from plotly import graph_objects as go
import cv2
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import imageio
import medmnist
import ipywidgets
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
```

#### Training

```
#Preprocess
#Resizing
path="static/dataset/"+fname
path2="static/training/"+fname
```



```

mm2 = PIL.Image.open(path).convert('L')
rz = mm2.resize((400,300), PIL.Image.ANTIALIAS)
rz.save(path2)
#noise filter
img = cv2.imread('static/training/'+fname)
dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 15)
fname2='ns_'+fname
#cv2.imwrite("static/training/"+fname2, dst)
###binarization
'''image = cv2.imread('static/training/'+fname)
original = image.copy()
kmeans = kmeans_color_quantization(image, clusters=4)
# Convert to grayscale, Gaussian blur, adaptive threshold
gray = cv2.cvtColor(kmeans, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (3,3), 0)
thresh = cv2.adaptiveThreshold(blur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV,21,2)
# Draw largest enclosing circle onto a mask
mask = np.zeros(original.shape[:2], dtype=np.uint8)
cnts = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
cnts = sorted(cnts, key=cv2.contourArea, reverse=True)
for c in cnts:
((x, y), r) = cv2.minEnclosingCircle(c)
cv2.circle(image, (int(x), int(y)), int(r), (36, 255, 12), 2)
cv2.circle(mask, (int(x), int(y)), int(r), 255, -1)
break
# Bitwise-and for result
result = cv2.bitwise_and(original, original, mask=mask)
result[mask==0] = (0,0,0)
cv2.imwrite("static/training/bin_"+fname, thresh)'''
###RPN - Segment
img = cv2.imread('static/training/'+fname)
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(gray,0,255,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
kernel = np.ones((3,3),np.uint8)
opening = cv2.morphologyEx(thresh,cv2.MORPH_OPEN,kernel, iterations = 2)
# sure background area
sure_bg = cv2.dilate(opening,kernel,iterations=3)
# Finding sure foreground area

```

```
dist_transform = cv2.distanceTransform(opening,cv2.DIST_L2,5)
ret, sure_fg = cv2.threshold(dist_transform,0.7*dist_transform.max(),255,0)
# Finding unknown region
sure_fg = np.uint8(sure_fg)
segment = cv2.subtract(sure_bg,sure_fg)
img = Image.fromarray(img)
segment = Image.fromarray(segment)
path3="static/training/sg/sg_"+fname
segment.save(path3)
image = cv2.imread(path2)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
edged = cv2.Canny(gray, 50, 100)
image = Image.fromarray(image)
edged = Image.fromarray(edged)
#Feature extraction- Feature Fusion Neural Network
def FeatureFusion():
train_datagen = ImageDataGenerator(validation_split=0.15)
test_datagen = ImageDataGenerator(validation_split=0.15)
train_data = train_datagen.flow_from_directory(
r'C:\Users\mhfar\Desktop\Trunk\Data',
color_mode="rgb",
batch_size=32,
class_mode='categorical',
target_size=(100, 100),
shuffle=False,
seed=42,
subset='training')
test_data = train_datagen.flow_from_directory(
r'C:\Users\mhfar\Desktop\Trunk\Data',
color_mode="rgb",
batch_size=32,
class_mode='categorical',
target_size=(100, 100),
shuffle=False,
seed=42,
subset='validation')
train_x=np.concatenate([train_data.next()[0] for i in range(train_data.__len__())])
train_y=np.concatenate([train_data.next()[1] for i in range(train_data.__len__())])
test_x=np.concatenate([test_data.next()[0] for i in range(test_data.__len__())])
```

```

test_y=np.concatenate([test_data.next()[1] for i in range(test_data.__len__())])

train_y = np.argmax(train_y, axis = 1)

test_y = np.argmax(test_y, axis = 1)

return train_x, train_y, test_x, test_y

def NBest(data, label, Num):

chi2selection = SelectKBest(chi2, k=Num)

newdata = chi2selection.fit_transform(data, label)

return newdata

def GABOR_Features(img):

histograms = []

for theta in range(0, 4):

theta = deepcopy(theta/4. * np.pi)

for sigma in (2, 4):

for lambda_ in np.arange(np.pi / 4, np.pi, np.pi / 4.):

for gamma in (0.05, 0.5):

kernel__ = cv.getGaborKernel((8, 8), sigma, theta, lambda_, gamma, 0, ktype=cv.CV_32F)

filtered = cv.filter2D(img, ddepth=4, kernel= kernel__)

hist = cv.calcHist([np.float32(filtered)], [0], None, [256], [0,256]).reshape(-1)

histograms.append(hist)

return np.reshape(histograms, (-1))

#Classification

def CNN():

#Lets start by loading the Cifar10 data

(X, y), (X_test, y_test) = cifar10.load_data()

#Keep in mind the images are in RGB

#So we can normalise the data by diving by 255

#The data is in integers therefore we need to convert them to float first

X, X_test = X.astype('float32')/255.0, X_test.astype('float32')/255.0

#Then we convert the y values into one-hot vectors

#The cifar10 has only 10 classes, thats is why we specify a one-hot

#vector of width/class 10

y, y_test = u.to_categorical(y, 10), u.to_categorical(y_test, 10)

#Now we can go ahead and create our Convolution model

model = Sequential()

#We want to output 32 features maps. The kernel size is going to be

#3x3 and we specify our input shape to be 32x32 with 3 channels

#Padding=same means we want the same dimensional output as input

#activation specifies the activation function

model.add(Conv2D(32, (3, 3), input_shape=(32, 32, 3), padding='same',

```

```

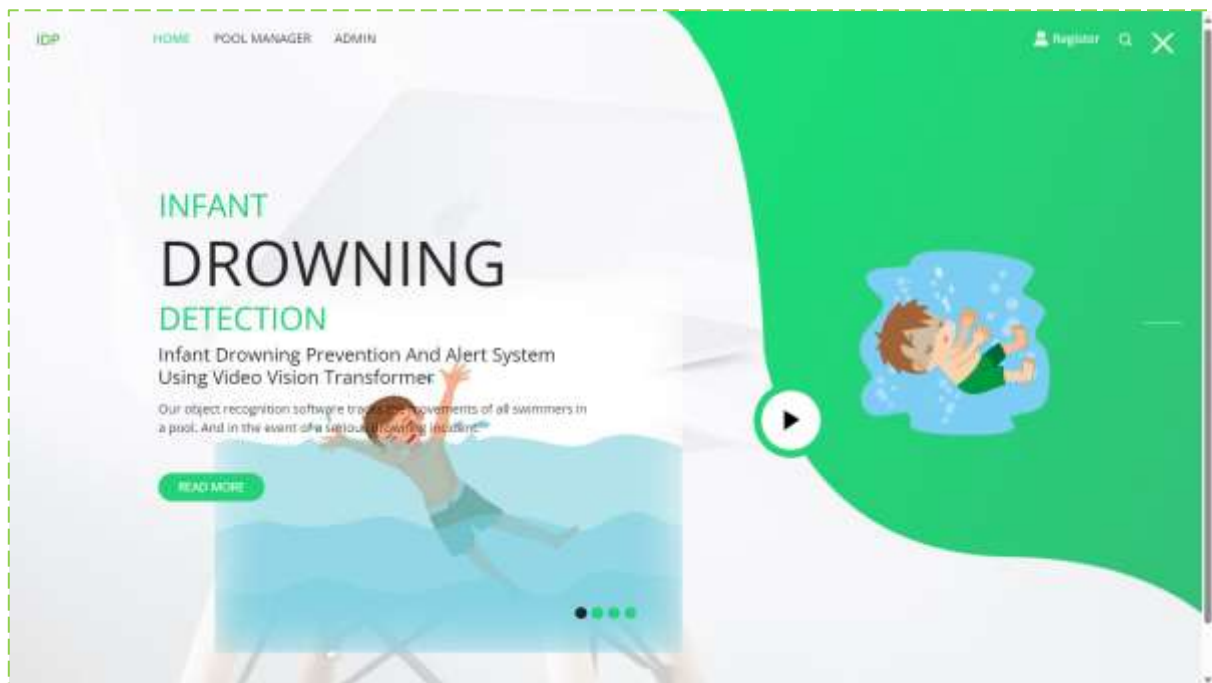
activation='relu'))
#20% of the nodes are set to 0
model.add(Dropout(0.2))
#now we add another convolution layer, again with a 3x3 kernel
#This time our padding=valid this means that the output dimension can
#take any form
model.add(Conv2D(32, (3, 3), activation='relu', padding='valid'))
#maxpool with a kernel of 2x2
model.add(MaxPooling2D(pool_size=(2, 2)))
#In a convolution NN, we need to flatten our data before we can
#input it into the output/dense layer
model.add(Flatten())
#Dense layer with 512 hidden units
model.add(Dense(512, activation='relu'))
#this time we set 30% of the nodes to 0 to minimize overfitting
model.add(Dropout(0.3))
#Finally the output dense layer with 10 hidden units corresponding to
#our 10 classes
model.add(Dense(10, activation='softmax'))
#Few simple configurations
model.compile(loss='categorical_crossentropy',
optimizer=SGD(momentum=0.5, decay=0.0004), metrics=['accuracy'])
#Run the algorithm!
model.fit(X, y, validation_data=(X_test, y_test), epochs=25,
batch_size=512)
#Save the weights to use for later
model.save_weights("cifar10.hdf5")
#Finally print the accuracy of our model!
print("Accuracy: &2.f%%" % (model.evaluate(X_test, y_test)[1]*100))
#Live Monitoring - Video Vision Transformer
def preprocess():
frames = tf.image.convert_image_dtype(
frames tf.newaxis
tf.float32,
# Parse label
label = tf.cast(label, tf.float32)
return frames, label
def prepare_data_loader(
videos: np.ndarray,

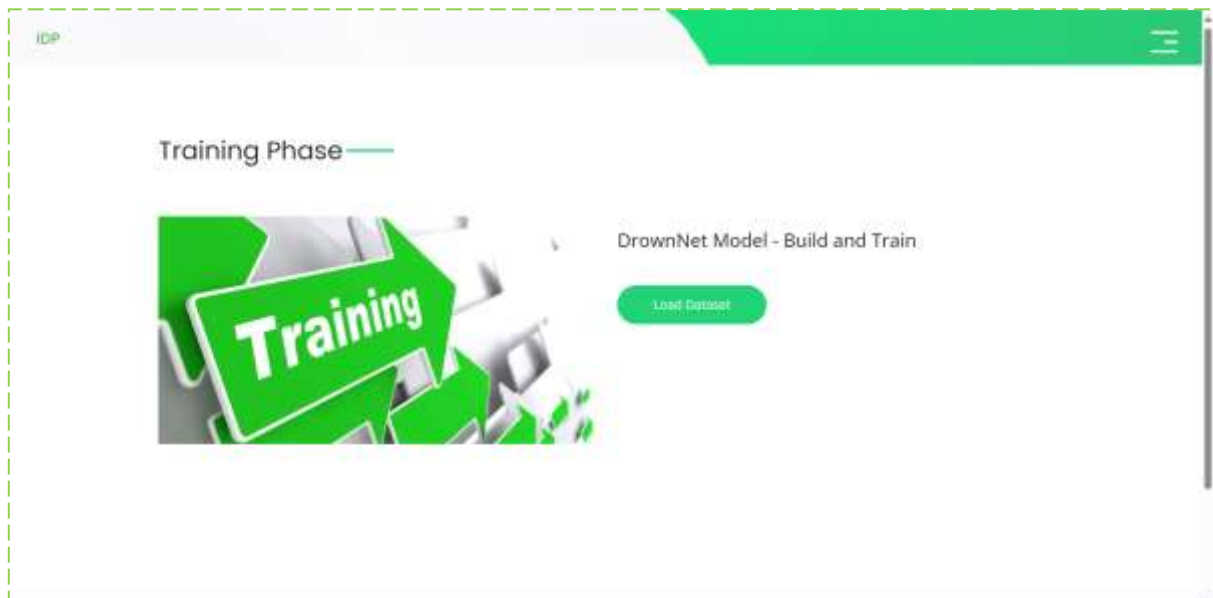
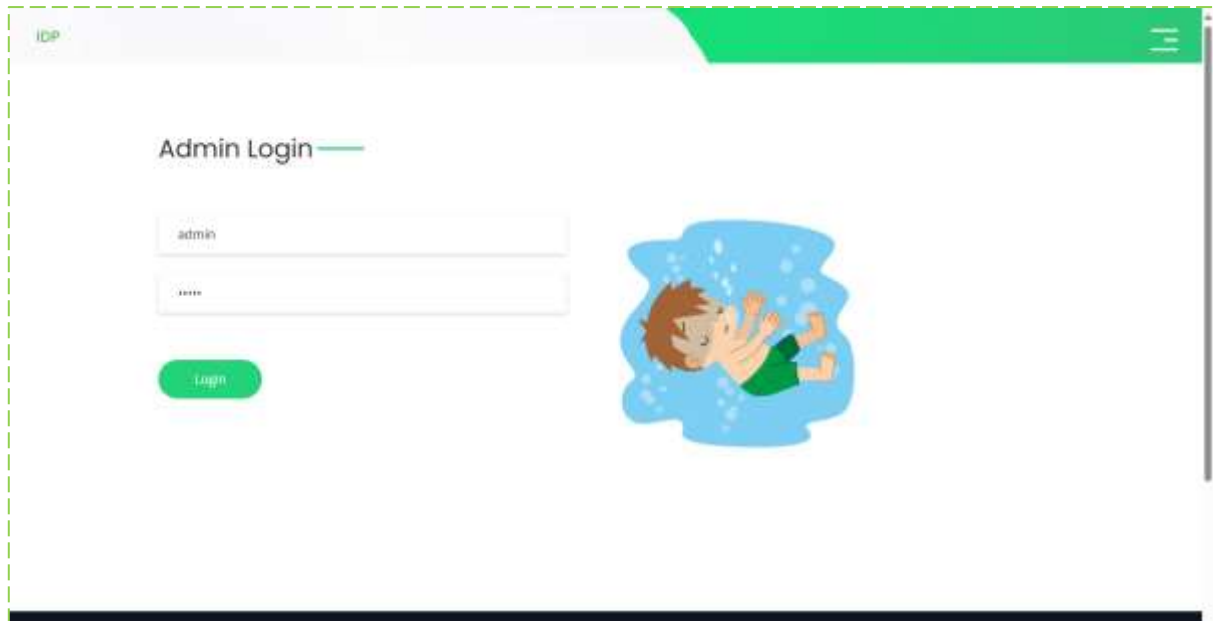
```

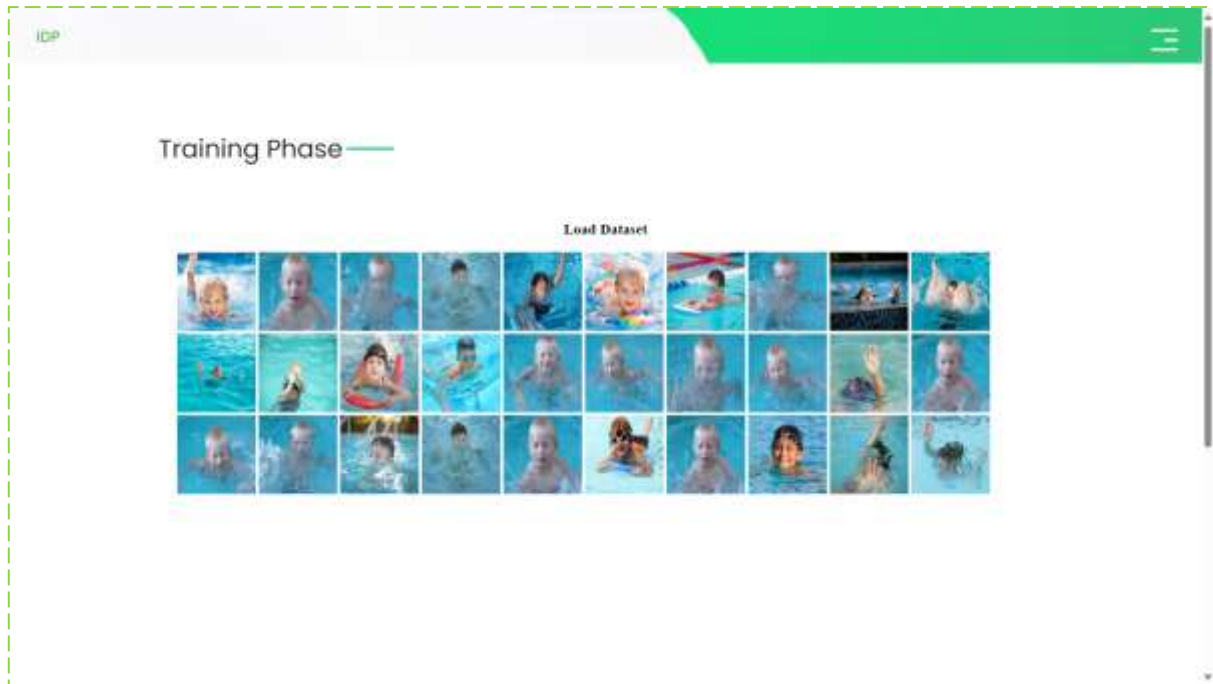
```
labels: np.ndarray,
loader_type: str = "train",
batch_size: int = BATCH_SIZE,
dataset = tf.data.Dataset.from_tensor_slices((videos, labels))
if loader_type == "train":
dataset = dataset.shuffle(BATCH_SIZE * 2)
dataloader = (
dataset.map(preprocess, num_parallel_calls=tf.data.AUTOTUNE)
.batch(batch_size)
.prefetch(tf.data.AUTOTUNE)
return dataloader
def create_vivit_classifier(
tubelet_embedder,
positional_encoder,
input_shape=INPUT_SHAPE,
transformer_layers=NUM_LAYERS,
num_heads=NUM_HEADS,
embed_dim=PROJECTION_DIM,
layer_norm_eps=LAYER_NORM_EPS,
num_classes=NUM_CLASSES,
# Get the input layer
inputs = layers.Input(shape=input_shape)
# Create patches.
patches = tubelet_embedder(inputs)
# Encode patches.
encoded_patches = positional_encoder(patches)
# Create multiple layers of the Transformer block.
for _ in range(transformer_layers):
# Layer normalization and MHSA
x1 = layers.LayerNormalization(epsilon=1e-6)(encoded_patches)
attention_output = layers.MultiHeadAttention(
num_heads=num_heads, key_dim=embed_dim // num_heads, dropout=0.1
)(x1, x1)
# Skip connection
x2 = layers.Add()([attention_output, encoded_patches])
# Layer Normalization and MLP
x3 = layers.LayerNormalization(epsilon=1e-6)(x2)
x3 = keras.Sequential(
layers.Dense(units=embed_dim * 4, activation=tf.nn.gelu),
```

```
layers.Dense(units=embed_dim, activation=tf.nn.gelu),  
(x3)  
encoded_patches = layers.Add()(x3, x2)  
# Layer normalization and Global average pooling.  
representation = layers.LayerNormalization(epsilon=layer_norm_eps)(encoded_patches)  
representation = layers.GlobalAvgPool1D()(representation)  
# Classify outputs.  
outputs = layers.Dense(units=num_classes, activation="softmax")(representation)  
# Create the Keras model.  
model = keras.Model(inputs=inputs, outputs=outputs)  
return model
```

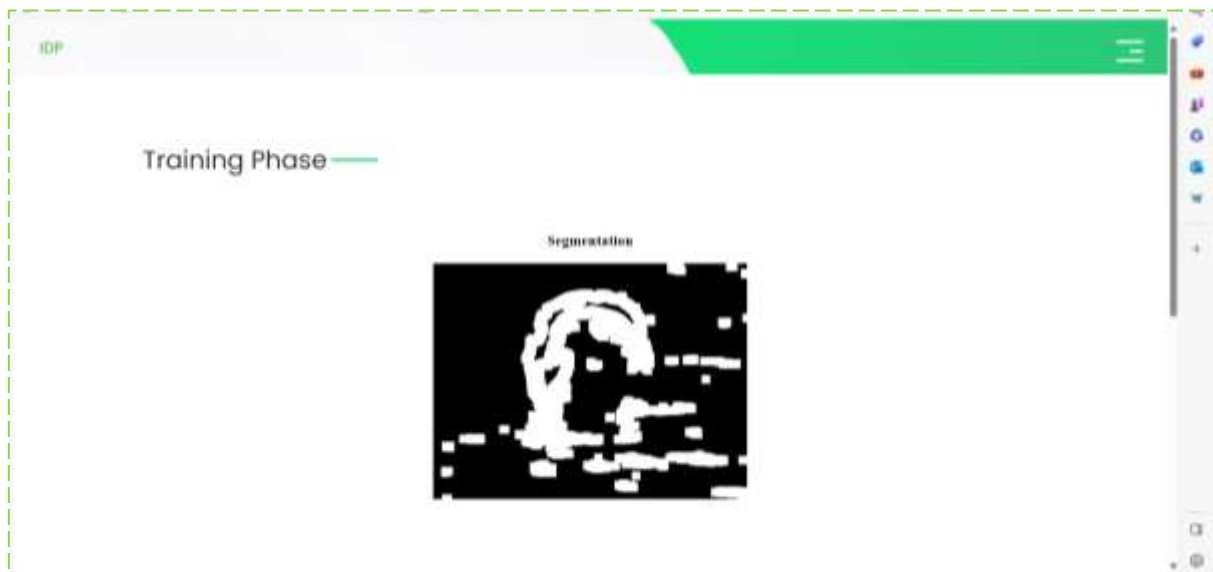
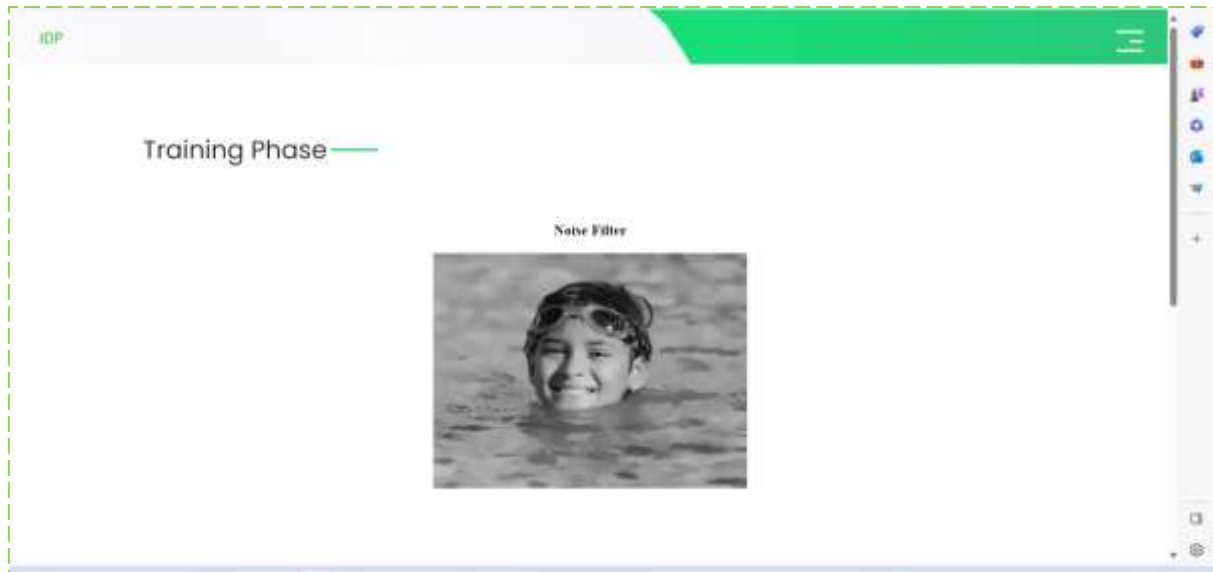
## SCREEN LAYOUTS

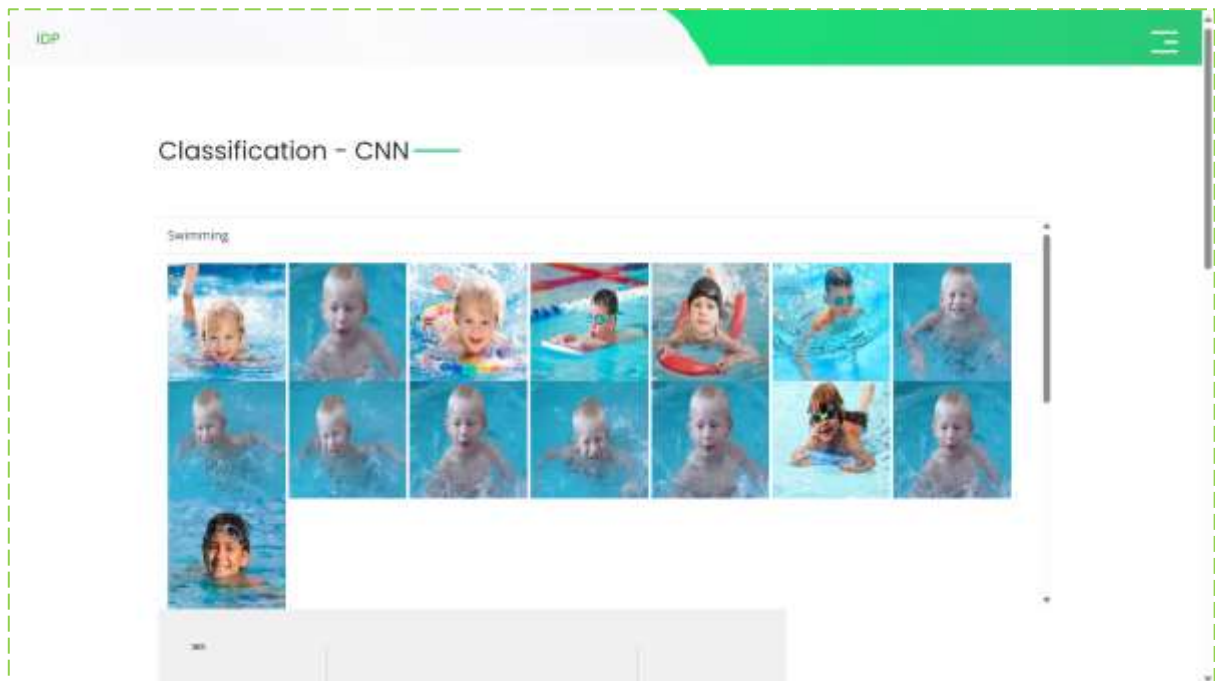













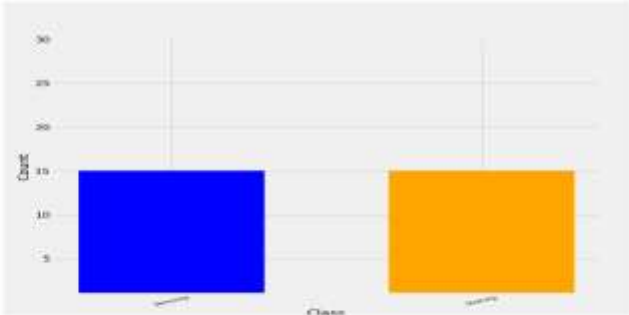
ICP

### Classification - CNN

Drowning



30



| Class   | Count |
|---------|-------|
| Class 1 | 15    |
| Class 2 | 15    |

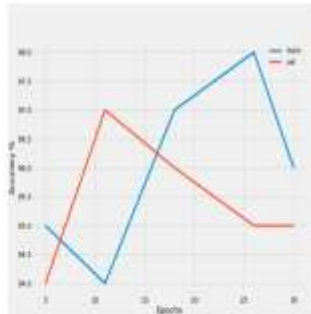
DrownNet Model - Build and Trained  
Performance Analysis



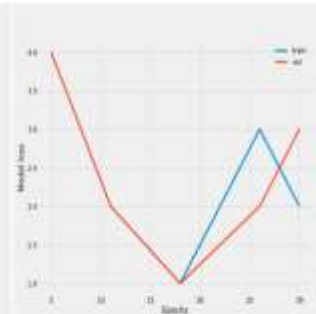
### DrownNet Model - Build and Trained

#### Performance Analysis

Model Accuracy




Model Loss




ICP

## Pool Manager - Registration



IDP

### Pool Manager Login



IDP

### Caretaker Details

Child Name

Caretaker Name

Caretaker Mobile No.

