ELLIE'S ADVENTURES

A gamified application to improve **communication**, **emotional**, **verbal** and **motor skills** in children with ASD.

SLIIT FACULTY OF COMPUTING

RP 24-25J-114 *2024-12-05*

PROJECT SUPERVISION

DR. KALPANI MANATHUNGA

Head, Department of Computer Science and Software Engineering

PROF. SAMANTHA THELIJJAGODA

Pro Vice-Chancellor (SLIIT Research & International)

DR. ASIRI HEWAMALAGE

National Program Manager, Child Development and Special Needs, Family Health Bureau, Ministry of Health

MR. SAMITHA VIDANAARACHCHI

Former Lecturer - SLIIT / PhD Researcher at Murdoch University, Australia

PROJECT TEAM

UMAIRA MARIKKAR

Year 4 Undergraduate
SLIIT

THEJANI MALLAWAARACHCHI

Year 4 Undergraduate
SLIIT

JATHURSHAN MANISTAR

Year 4 Undergraduate
SLIIT

PAWAN SENPURA

Year 4 Undergraduate
SLIIT

03 INTRODUCTION

What is Ellie’s Adventures?

A Gamified Mobile App

Customizable 3D avatar “Ellie”

Comprises serious-game exercises addressing several focus areas

Aims to engage the child to improve in these areas

Research Conditions

Autism is a spectrum of disorders, this research will be conducted for a selected community of the spectrum

1. Children between the age of 3 to 12.

2. Activities to be done under parental supervision.

3. Children who can speak but not comprehensively.

80 % [3]
speech development

87 % [4]
Motor Skills

42 - 50 % [3]
Mental retardation

40.9 % [1]
language comprehension

Every 1 in 100 [1]

7.4% OF CHILDREN [2]

EVERY 1 IN 93 [2]

Affected from ASD

ETHNOGRAPHIC STUDIES

Christine Activity Center
Malabe

ETHNOGRAPHIC STUDIES

- Family Health Bureau
- Namaste Program for Autism Awareness in Sri Lanka

03 MAIN OBJECTIVES

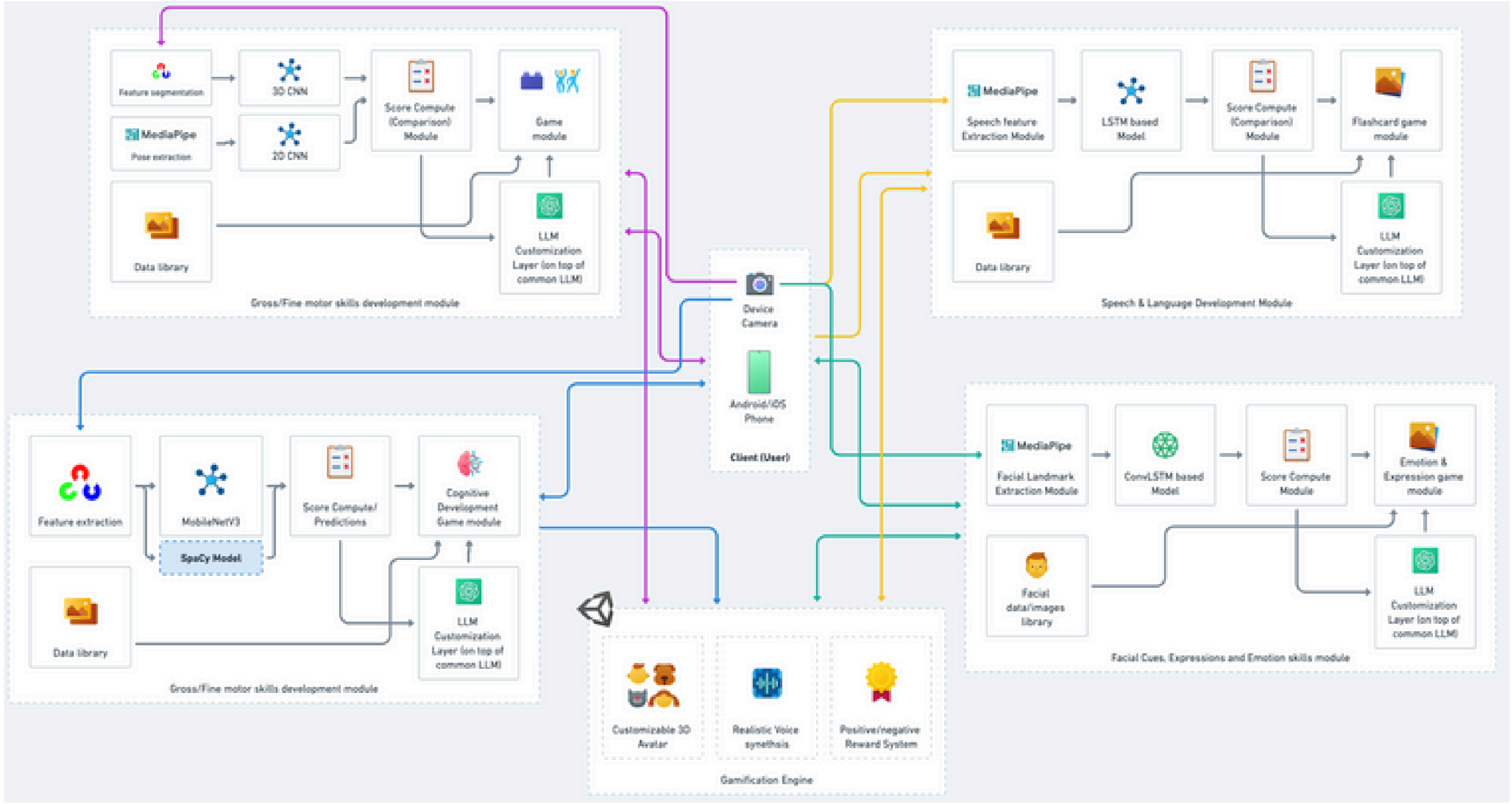
SPEECH AND LANGUAGE DEVELOPMENT

COGNITIVE AND VERBAL SKILL DEVELOPMENT

FINE AND GROSS MOTOR SKILL DEVELOPMENT

FACIAL CUES AND RECOGNITION EXPRESSION

SYSTEM ARCHITECTURE



IMPROVING GROSS & FINE MOTOR SKILLS

Mallawaarachchci T. D. R.
IT21282836

[Software Engineering]

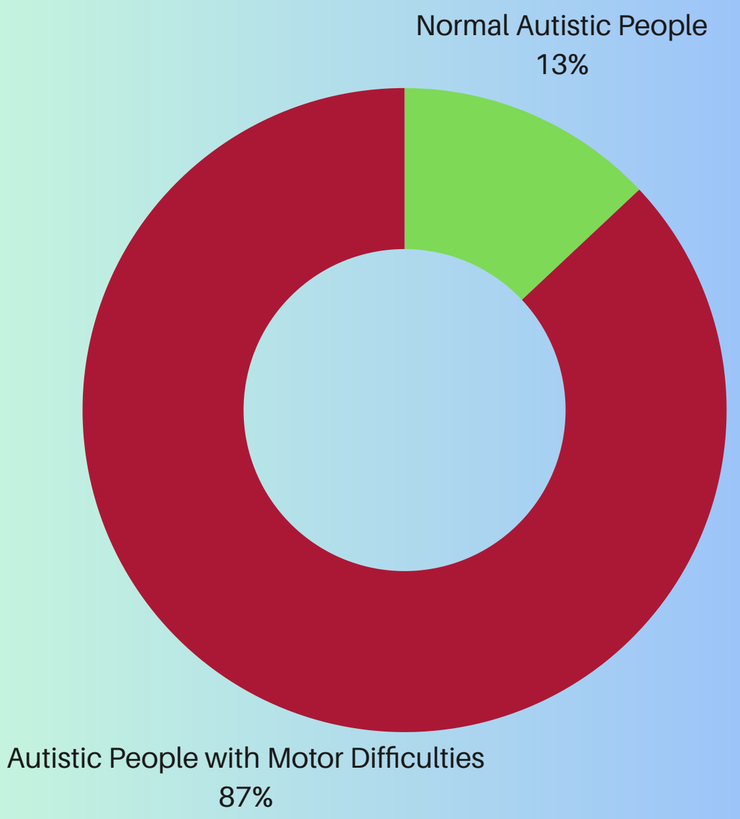
Background

Children with ASD often experience **delays in motor skills development**, affecting their daily activities and overall growth.

Research indicates ASD children has significant **delays in gross motor skills (6.7%) and fine motor skills (38.5%) compared to typically developing children** [2]

Motor skill activities can stimulate **cognitive development** and **improve attention, planning, and problem-solving abilities.**

Addressing both **gross and fine motor skills** is essential for comprehensive motor skill development.



According to the latest estimate **87% of the Autistic people** face some sort of **motor difficulty**. [1]

Research Questions

How to improve gross & fine motor skills of a child with autism ?

1. How to identify and measure a child's **current motor skill abilities** at home?

2. What **milestones and criteria should be used to assess** the child's motor skill progress?

3. What **activities** or games can effectively target **both gross and fine motor skill development** at home?

4. What **methodologies** can be used to enhance motor skills development?

5. How to provide the **necessary guidance** to support the child's motor skill improvements?

Main and Specific Objectives

Improving Motor Skills of Children with ASD

- *Gross Motor Skills*
- *Fine Motor Skills*

Train the model to identify the actions performed by the child.

Identify and evaluate the child's ability to mimic the given exercises correctly.

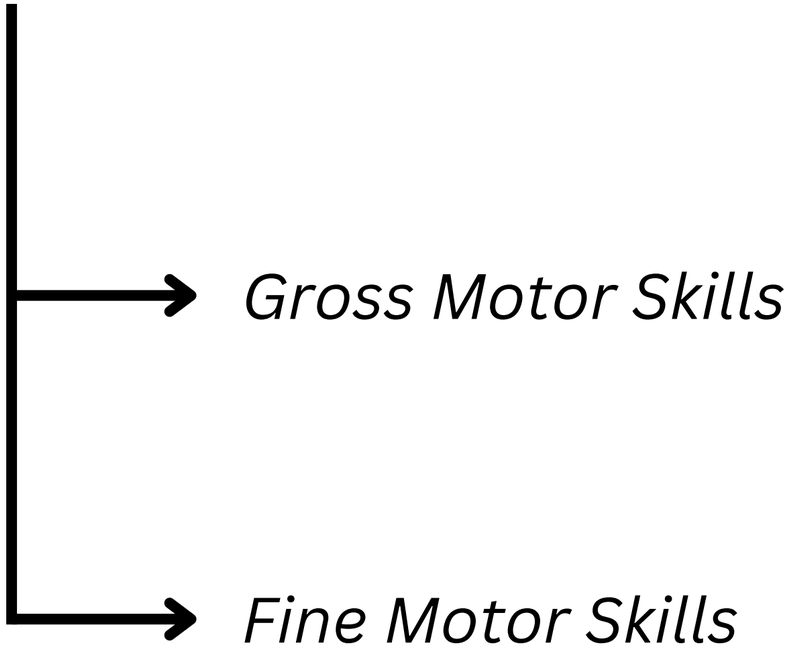
Provide feedback and encouragement based on the performance.

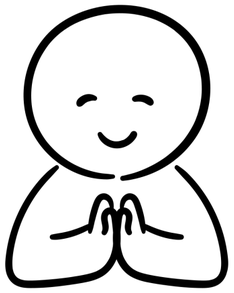
Train the YOLO model to identify objects correctly.

Evaluate ability of the child to create the color pattern and give the feedback.

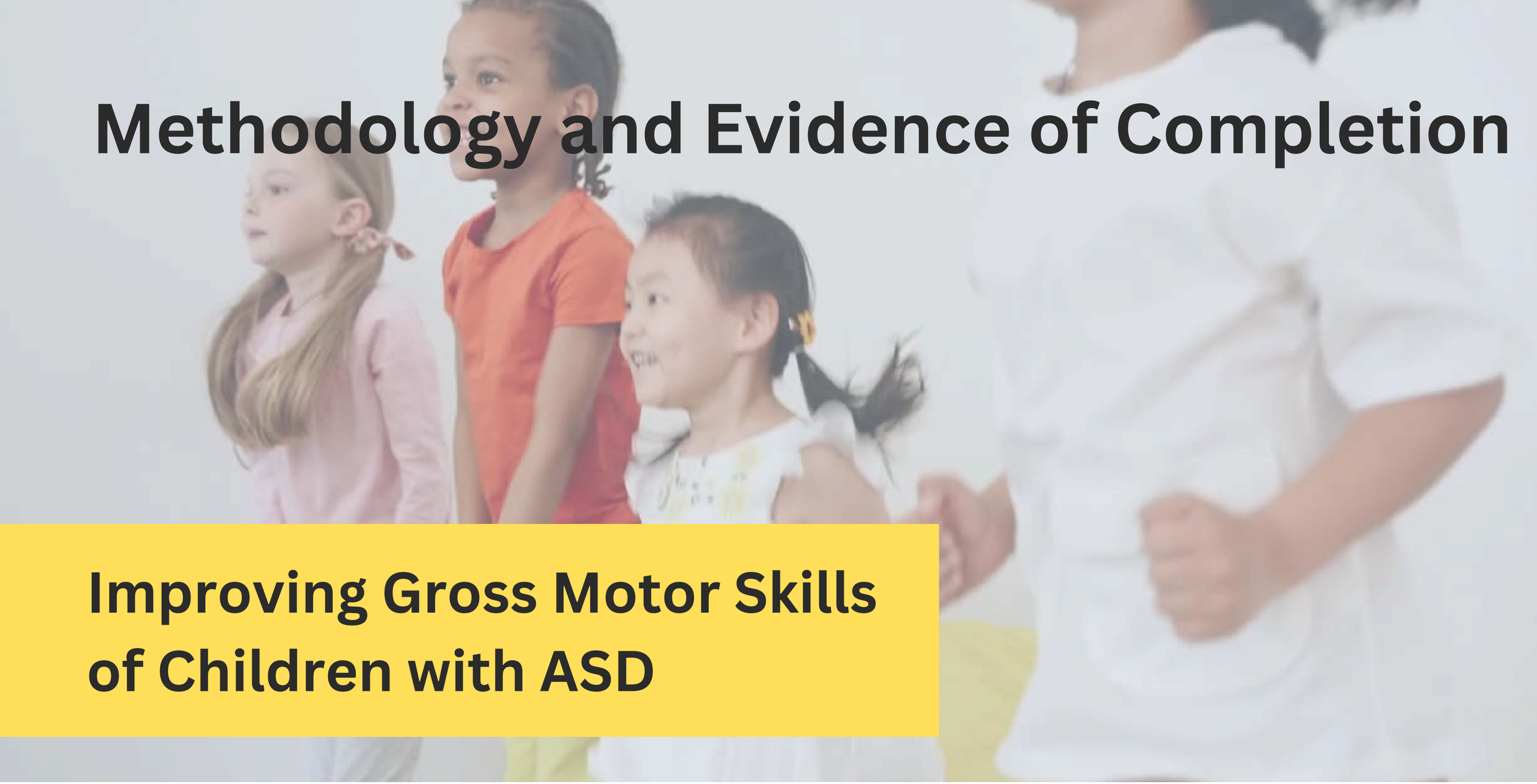
Methodology and Evidence of Completion

Improving Motor Skills of Children with ASD



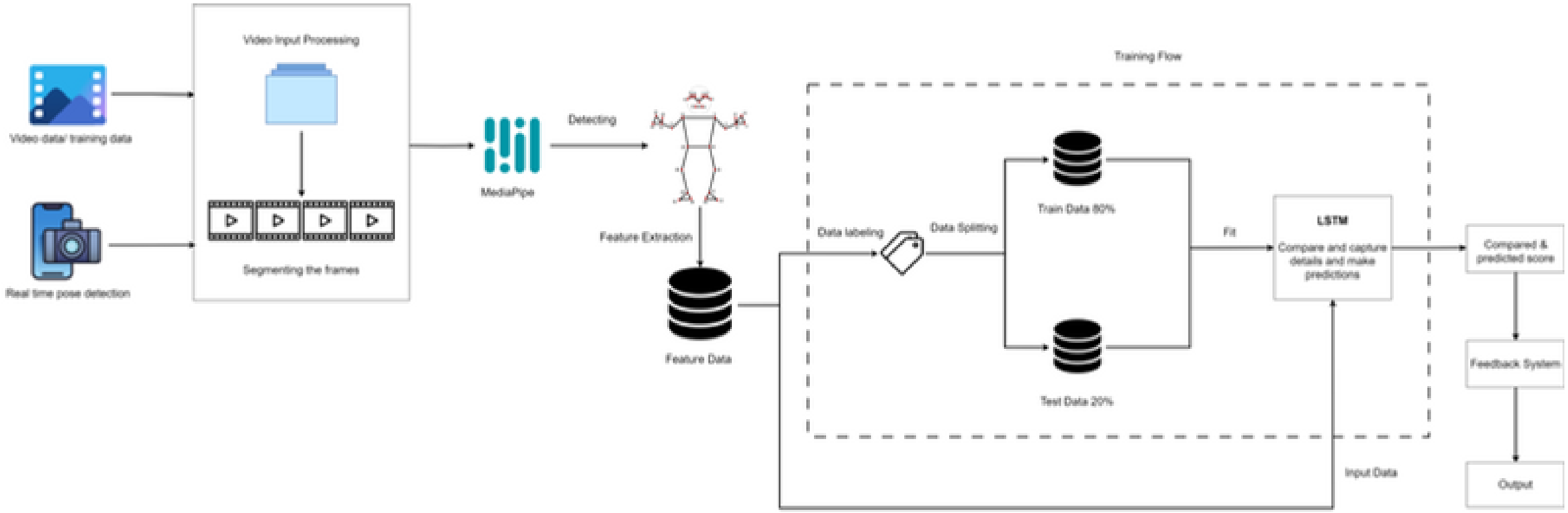


Methodology and Evidence of Completion



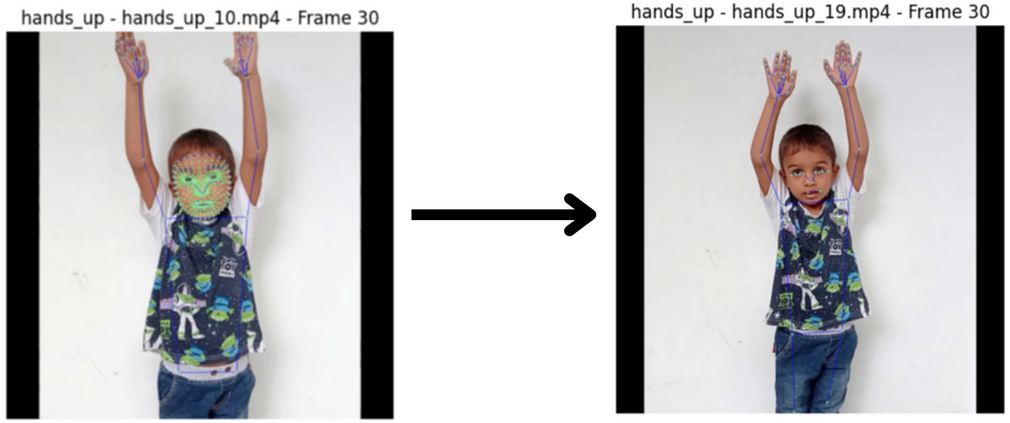
Improving Gross Motor Skills of Children with ASD

System Diagram



Objective 01 & 02 - Train the model and evaluate the child's ability to mimic the given exercises correctly.

Evidence : Preprocessed data for keypoint extraction



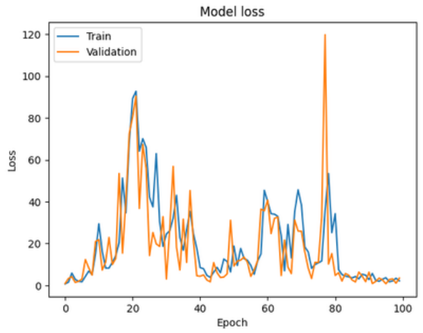
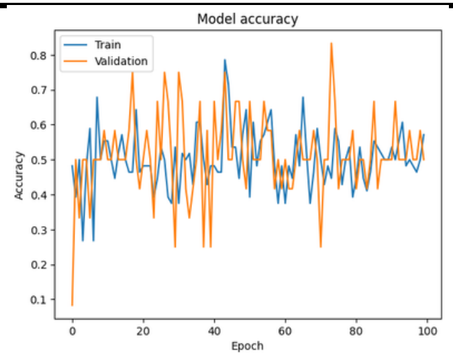
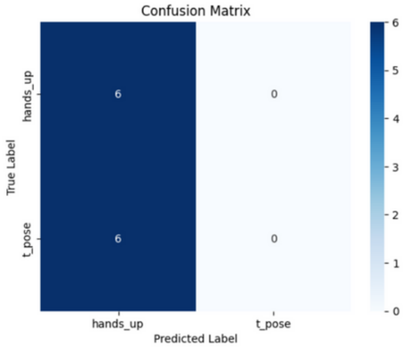
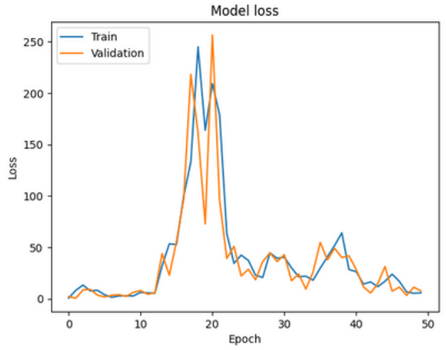
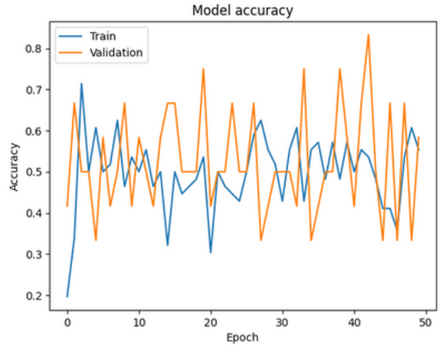
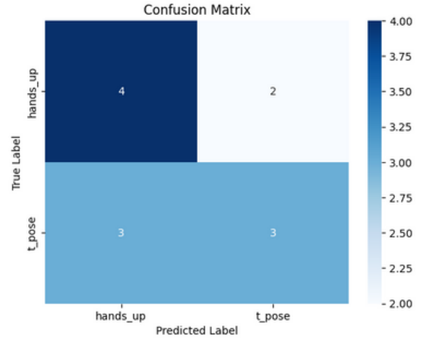

```
model = Sequential() # sequential API
model.add(LSTM(64, return_sequences=True, activation='relu', input_shape=(sequence_length, 1662)))
model.add(LSTM(128, return_sequences=True, activation='relu'))
model.add(LSTM(64, return_sequences=False, activation='relu')) # not returning sequences into dense
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

```
model = Sequential() # sequential API
model.add(LSTM(64, return_sequences=True, activation='relu', input_shape=(sequence_length, 258)))
model.add(LSTM(128, return_sequences=True, activation='relu'))
model.add(LSTM(64, return_sequences=False, activation='relu')) # not returning sequences into dens
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

Objective 01 & 02 Contd.

Train the model and evaluate the child's ability to mimic the given exercises correctly.

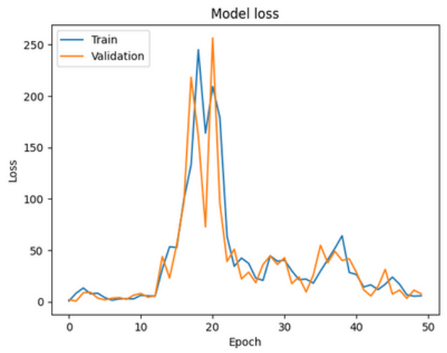
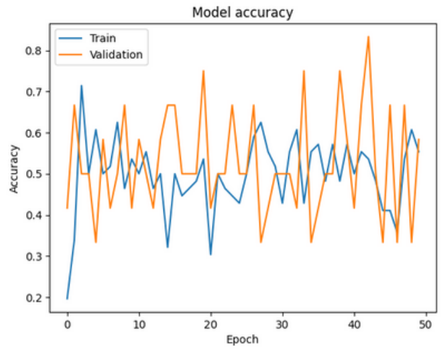
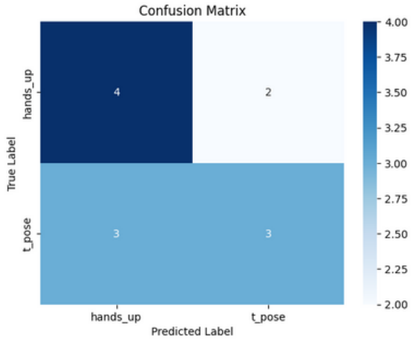
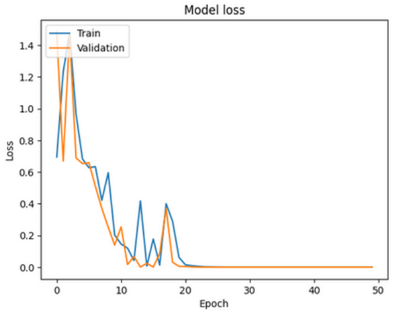
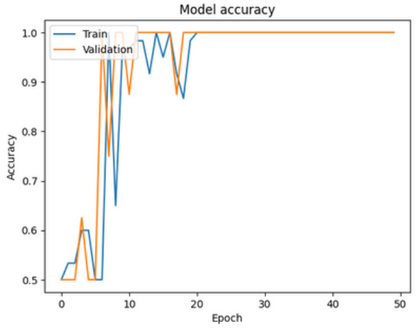
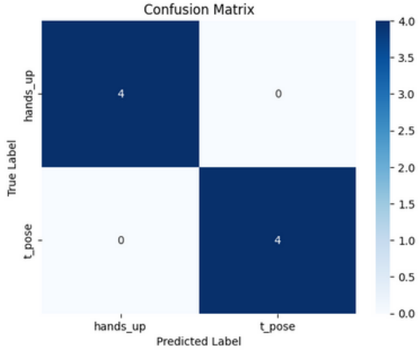
Evidence: Training the model with different dataset sizes, and epoch sizes

Epochs	No of Videos	Data Split	Loss Graph	Accuracy Graph	Confusion Matrix
100	34 videos per action	Training - 56 Testing - 12			
50	34 videos per action	Training - 56 Testing - 12			

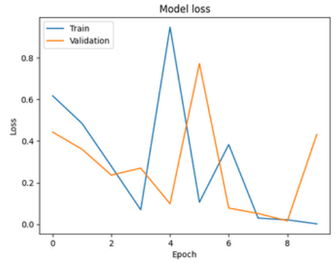
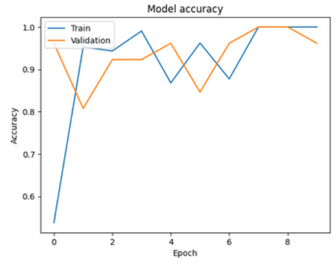
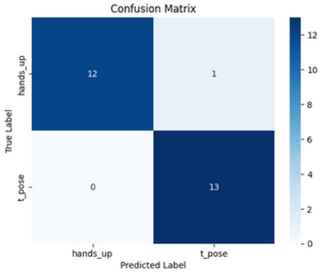
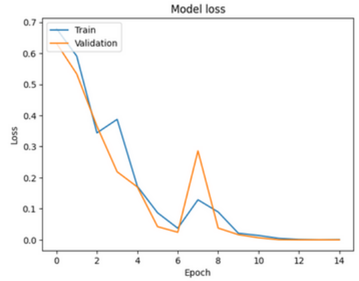
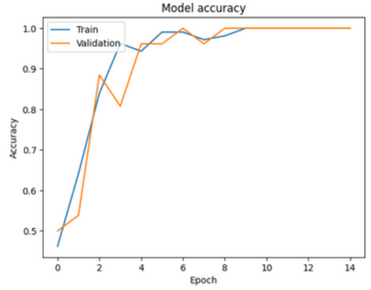
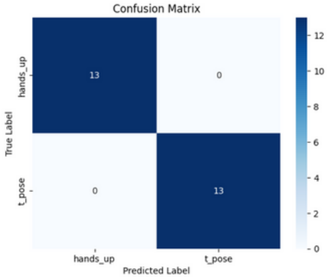
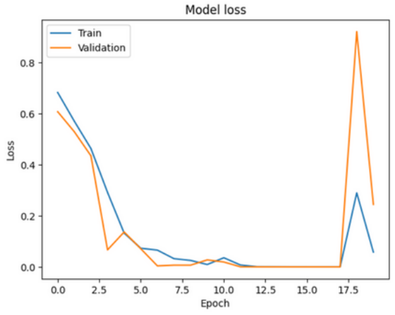
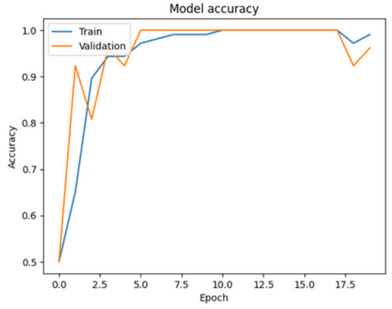
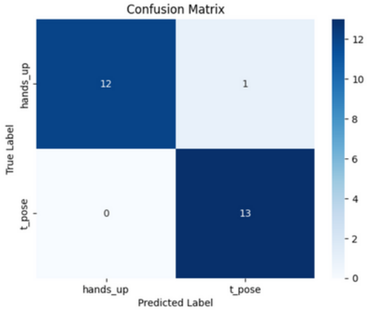
Objective 01 & 02 Contd.

Train the model and evaluate the child's ability to mimic the given exercises correctly.

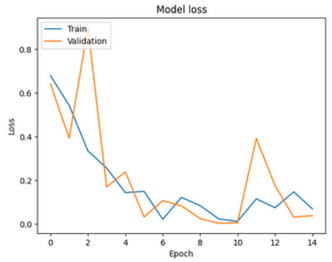
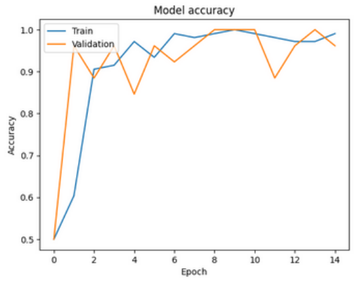
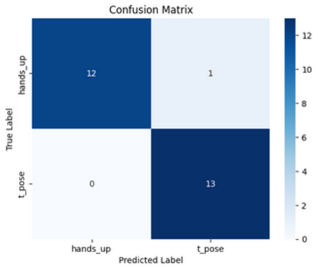
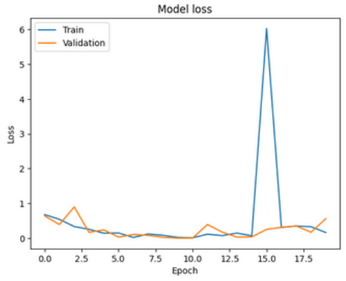
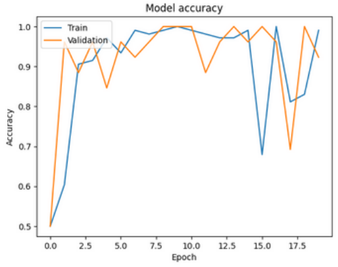
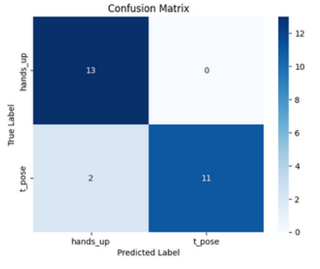
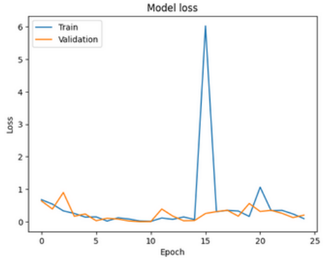
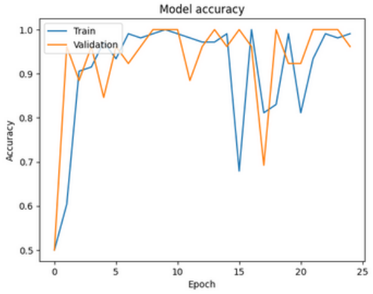
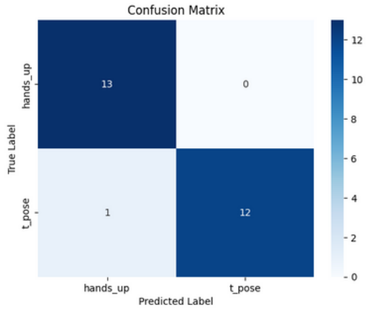
Evidence: Training the model with different dataset sizes, and epoch sizes

Epochs	No of Videos	Data Split	Loss Graph	Accuracy Graph	Confusion Matrix
50	34 videos per action	Training - 56 Testing - 12			
50	34 videos per action	Training - 60 Testing - 08			

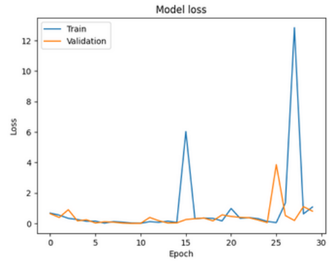
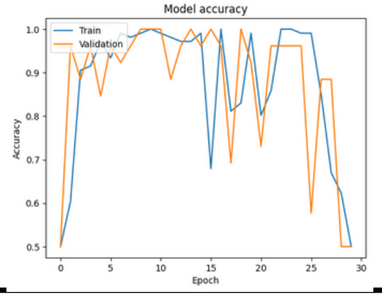
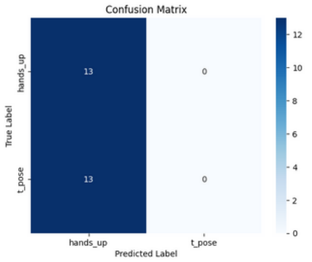
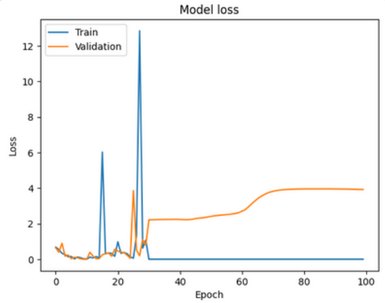
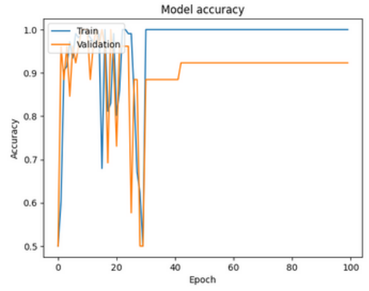
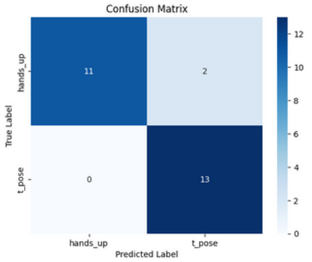
Objective 01 & 02 Contd.

Epochs	No of Videos	Data Split	Loss Graph	Accuracy Graph	Confusion Matrix
10	66 videos per action	Train - 106 Test - 26			
15	66 videos per action	Train - 106 Test - 26			
20	66 videos per action	Train - 106 Test - 26			

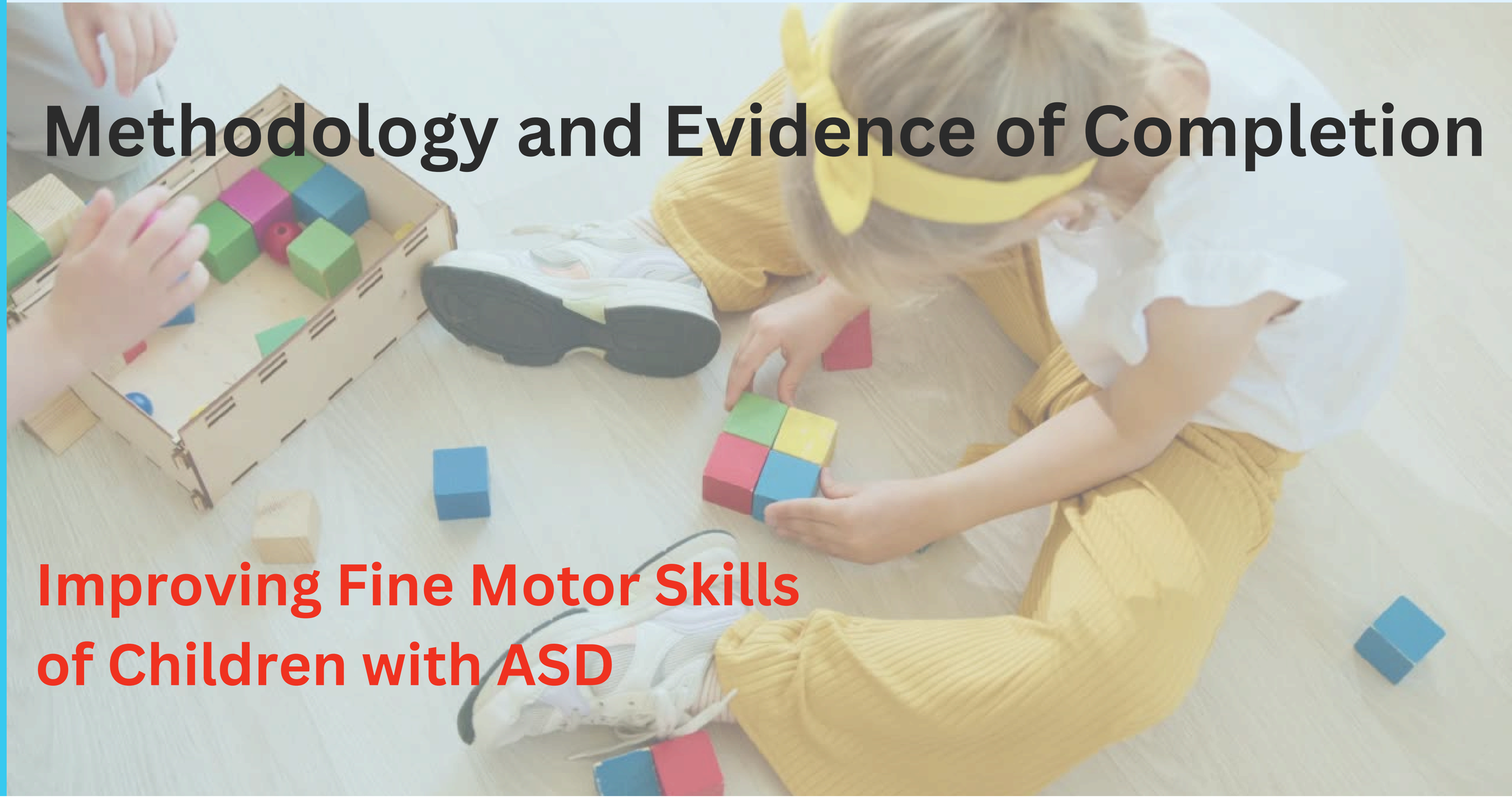
PP2 Ref

Epochs	No of Videos	Data Split	Loss Graph	Accuracy Graph	Confusion Matrix
15	66 videos p.a. 33 kp	Train - 106 Test - 26			
20	66 videos p.a. 33kp	Train - 106 Test - 26			
25	66 videos p.a. 33kp	Train - 106 Test - 26			

PP2 Ref

Epochs	No of Videos	Data Split	Loss Graph	Accuracy Graph	Confusion Matrix
30	66 videos p.a. 33 kp	Train - 106 Test - 26			
100	66 videos p.a. 33kp	Train - 106 Test - 26			
	66 videos p.a. 33kp	Train - 106 Test - 26			

Methodology and Evidence of Completion



Improving Fine Motor Skills of Children with ASD

Improving Fine Motor Skills of Children with ASD

Reasons to change the proposed approach:

- **To follow a more standardized way.**
- **To consider the safety of the child.**

First Box Of Color Tablets

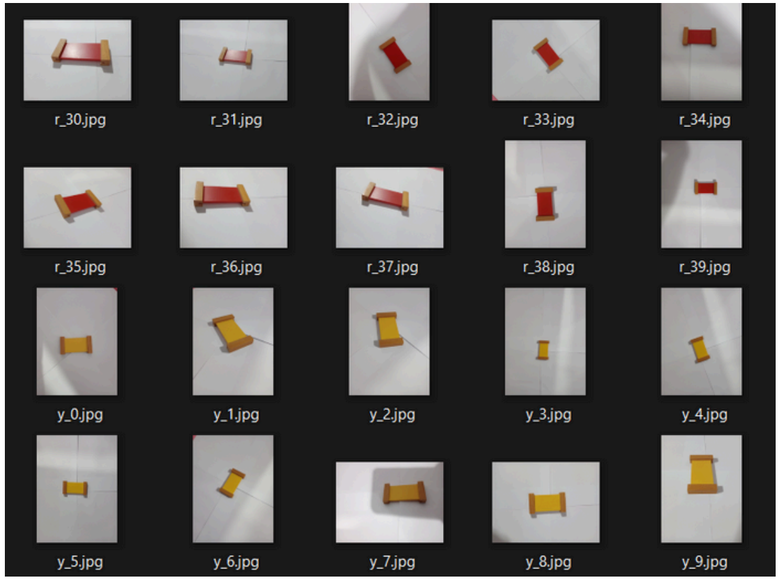
Third Box Of Color Tablets

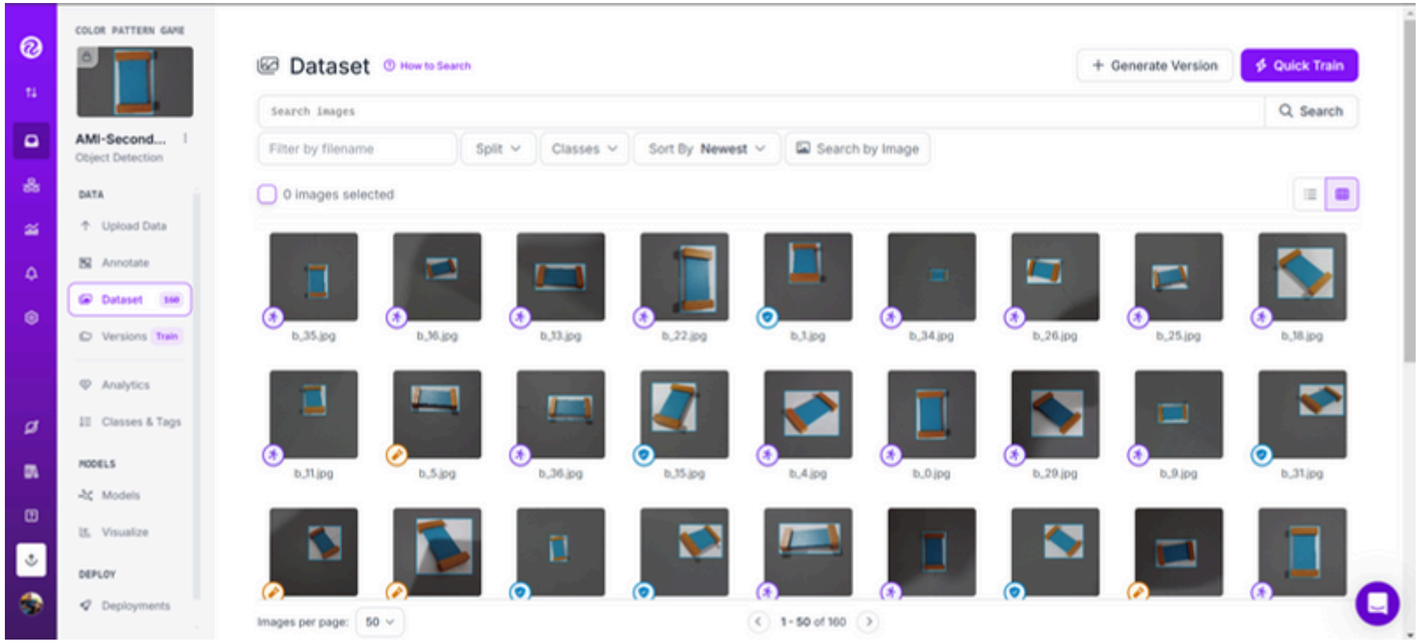
Second Box Of Color Tablets

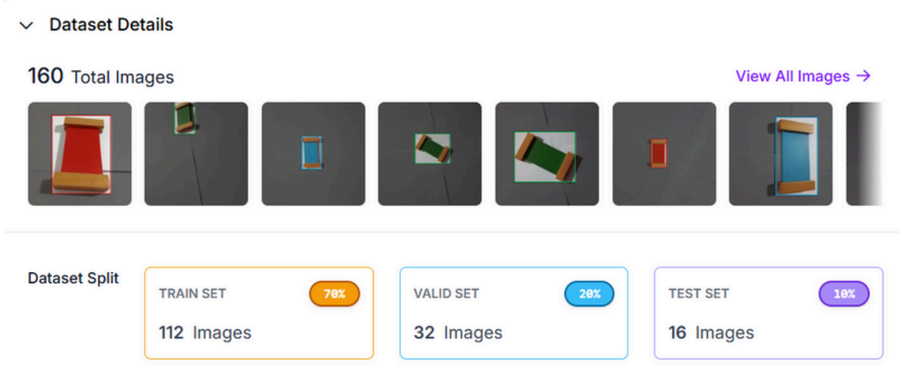
[3]

Objective 04 - Train the YOLO model to identify objects.

Evidence: Annotating collected AMI Second Box of Color Pallets dataset.







[4]

Objective 04 - Train the YOLO model to identify objects.

Evidence: Training the model with the annotated AMI Second Box of Color Pallets dataset.

```
%cd {HOME}

!yolo task=detect mode=train model=yolo11n.pt data={dataset.location}/data.yaml epochs=5 batch=16 imgsz=640 plots=True
```

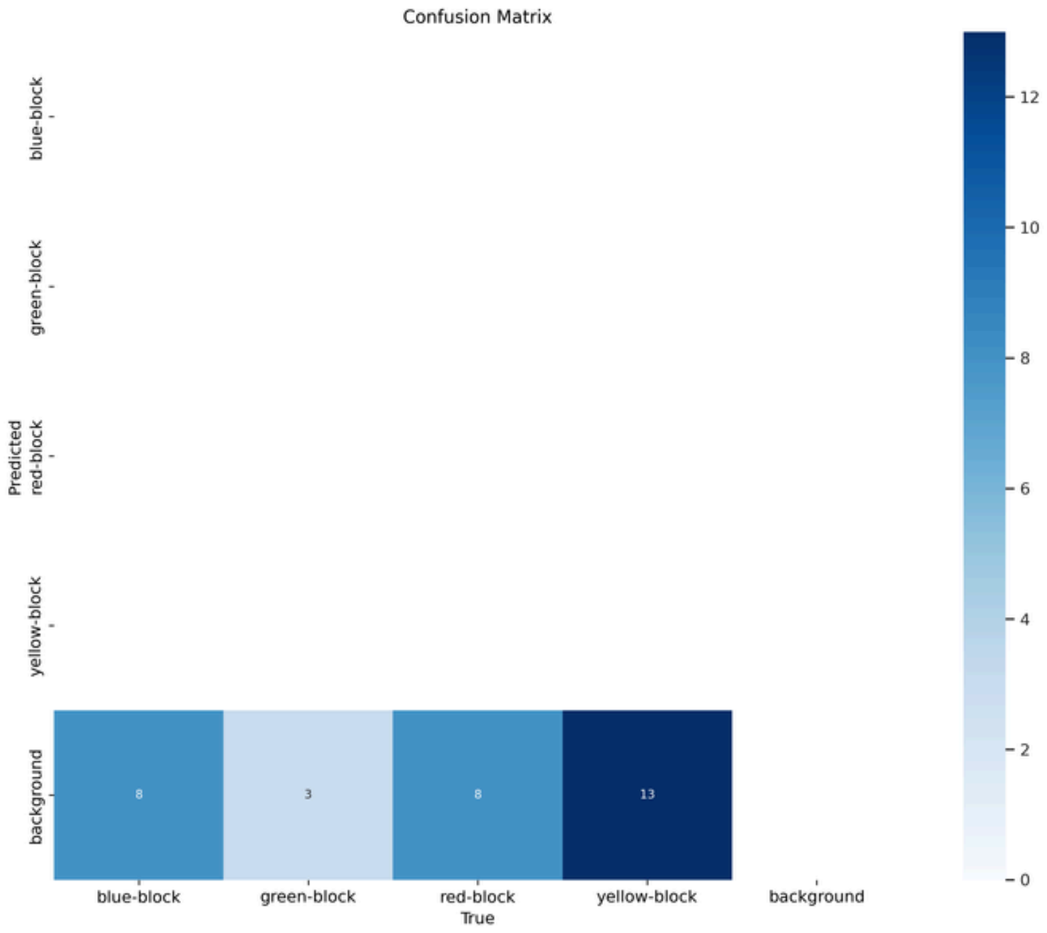
```
Logging results to runs/detect/train
Starting training for 5 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        1/5      2.54G      1.263      3.698      1.359         39        640: 100% 7/7 [00:08<00:00,  1.16s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:03<00:00,  3.88s/it]
                   all         32         32    0.00394      0.917      0.202       0.16

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        2/5      2.36G      0.889      3.434       1.17         29        640: 100% 7/7 [00:04<00:00,  1.63it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.94it/s]
                   all         32         32    0.00432          1      0.323      0.274

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        3/5      2.36G     0.7572      3.159      1.056         41        640: 100% 7/7 [00:04<00:00,  1.66it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.87it/s]
                   all         32         32    0.00599          1      0.415      0.381

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        4/5      2.36G     0.7086      3.043      1.046         39        640: 100% 7/7 [00:02<00:00,  2.54it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.59it/s]
                   all         32         32    0.00825          1      0.364      0.331

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        5/5      2.36G      0.646      2.875     0.9964         39        640: 100% 7/7 [00:03<00:00,  2.08it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.01it/s]
                   all         32         32     0.0188          1      0.384      0.361
```

5 epochs with 160 images
The modal didn't identify any block.

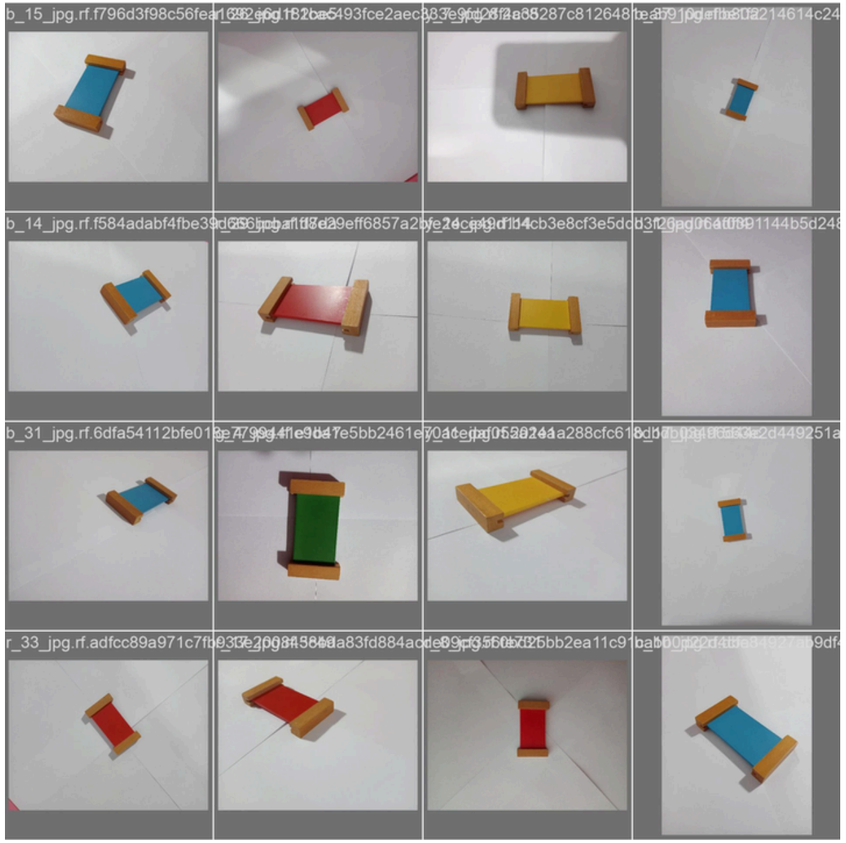
```
5 epochs completed in 0.012 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train/weights/best.pt, 5.5MB

Validating runs/detect/train/weights/best.pt...
Ultralytics 8.3.39 🚀 Python-3.10.12 torch-2.5.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLO11n summary (fused): 238 layers, 2,582,932 parameters, 0 gradients, 6.3 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.64it/s]
                   all         32         32    0.00599          1      0.399      0.367
            blue-block          8          8    0.00259          1      0.413      0.391
           green-block          3          3    0.00282          1      0.123      0.107
             red-block          8          8    0.00171          1      0.468      0.415
          yellow-block         13         13     0.0168          1      0.595      0.554
Speed: 0.6ms preprocess, 2.8ms inference, 0.0ms loss, 2.9ms postprocess per image
Results saved to runs/detect/train
```

Objective 04 - Train the YOLO model to identify objects.

Evidence: Training the model with the annotated AMI Second Box of Color Pallets dataset.



5 epochs with 160 images
The modal didn't identify any block.



Objective 04 - Train the YOLO model to identify objects.

Evidence: Training the model with the annotated AMI Second Box of Color Pallets dataset.

```
%cd {HOME}

# initial model train
# !yolo task=detect mode=train model=yolo11n.pt data={dataset.location}/data.yaml epochs=5 batch=16 imgsz=640 plots=True

# continue training after the initial train
!yolo task=detect mode=train model={HOME}/runs/detect/train/weights/best.pt data={dataset.location}/data.yaml epochs=20 batch=16 imgsz=640 plots=True
```

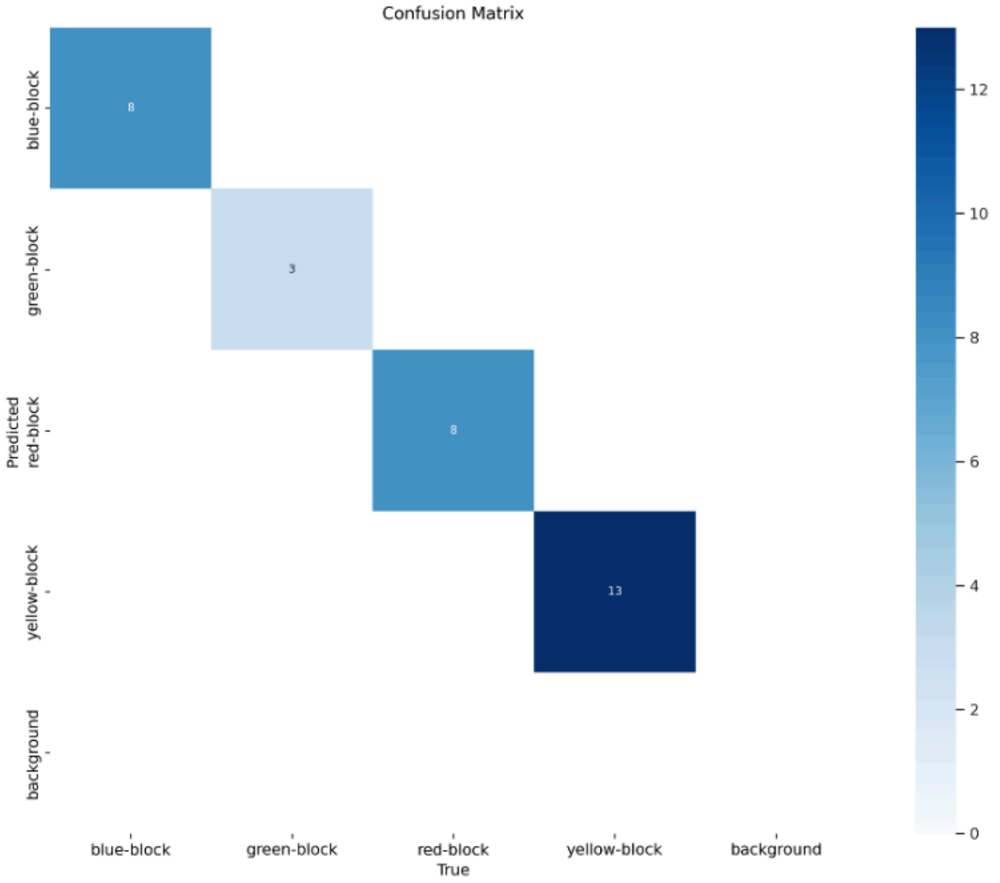
```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      13/20      2.36G     0.3753      1.606     0.8268         16        640: 100% 7/7 [00:02<00:00,  2.62it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.75it/s]
                   all         32         32      0.962      0.795       0.99      0.925

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      14/20      2.36G     0.3794       1.54     0.8367         16        640: 100% 7/7 [00:02<00:00,  2.67it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.07it/s]
                   all         32         32      0.924      0.984      0.995      0.927

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      15/20      2.36G     0.3564      1.396     0.8105         16        640: 100% 7/7 [00:03<00:00,  2.05it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.21it/s]
                   all         32         32      0.977          1      0.995      0.942

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      16/20      2.36G     0.3491       1.42     0.7859         16        640: 100% 7/7 [00:02<00:00,  2.54it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.13it/s]
                   all         32         32      0.974          1      0.995      0.944

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      17/20      2.36G     0.3303      1.365     0.8152         16        640: 100% 7/7 [00:02<00:00,  3.13it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.29it/s]
                   all         32         32      0.984      0.995      0.995      0.944

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      18/20      2.36G     0.3229      1.344     0.8178         16        640: 100% 7/7 [00:02<00:00,  2.91it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.27it/s]
                   all         32         32      0.973          1      0.995       0.95
```

```
20 epochs completed in 0.039 hours.
Optimizer stripped from runs/detect/train2/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train2/weights/best.pt, 5.5MB

Validating runs/detect/train2/weights/best.pt...
Ultralytics 8.3.39 🚀 Python-3.10.12 torch-2.5.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLO11n summary (fused): 238 layers, 2,582,932 parameters, 0 gradients, 6.3 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.58it/s]
                   all         32         32      0.982          1      0.995      0.971
           blue-block          8          8      0.982          1      0.995      0.967
          green-block          3          3      0.959          1      0.995      0.995
            red-block          8          8      0.995          1      0.995      0.954
         yellow-block         13         13      0.992          1      0.995       0.97
Speed: 0.5ms preprocess, 3.4ms inference, 0.0ms loss, 1.3ms postprocess per image
Results saved to runs/detect/train2
```

20 epochs with 160 images.

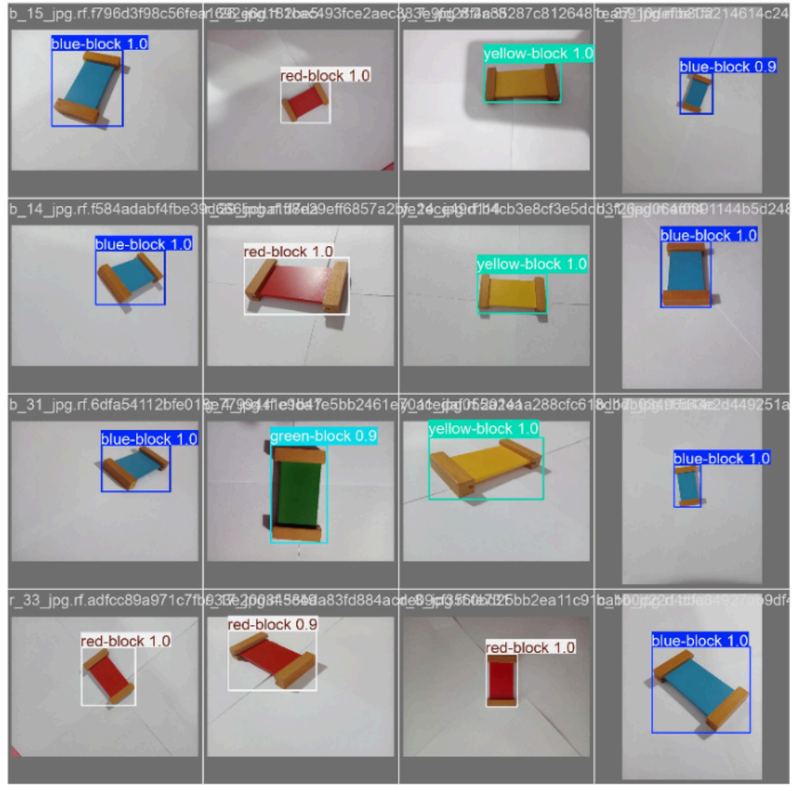
Objective 04 - Train the YOLO model to identify objects.

Evidence: Training the model with the annotated AMI Second Box of Color Pallets dataset.



20 epochs with 160 images

The modal identified the blocks in validation dataset.



Objective 04 - Train the YOLO model to identify objects.

Evidence: Training the model with the annotated AMI Second Box of Color Pallets dataset.

```
# previous model path
PREVIOUS_MODEL_PATH = "/content/drive/MyDrive/Thejani_Research_Y4/ColorPatternModel/v02-160_img_with_20_epochs/runs/detect/train/weights/best.pt"
# continue training after the initial train
!yolo task=detect mode=train model={PREVIOUS_MODEL_PATH} data={dataset.location}/data.yaml epochs=50 batch=16 imgsz=640 plots=True
# !yolo task=detect mode=train model={HOME}/runs/detect/train/weights/best.pt data={dataset.location}/data.yaml epochs=20 batch=16 imgsz=640 plots=True
```

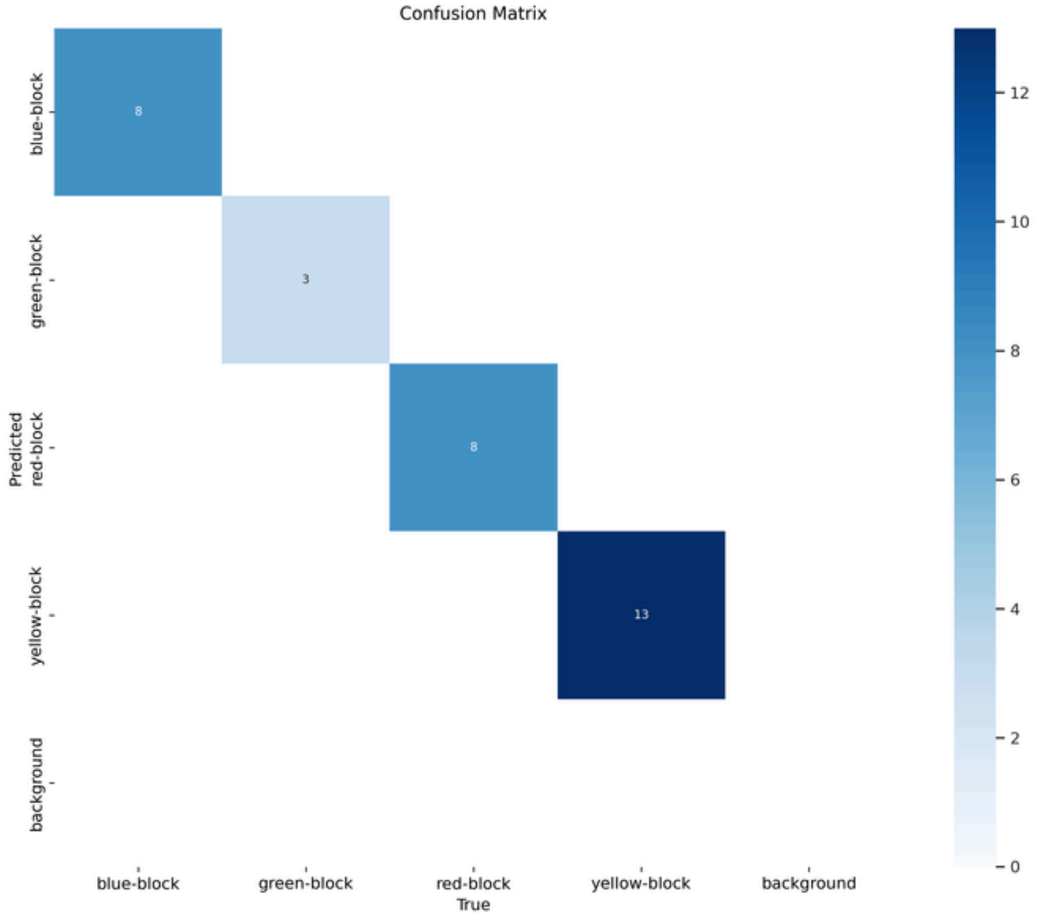
```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      43/50      2.36G      0.259     0.8216     0.7723         16        640: 100% 7/7 [00:04<00:00,  1.43it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.11it/s]
                   all         32         32       0.99          1      0.995      0.974

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      44/50      2.36G     0.2689      0.833     0.7872         16        640: 100% 7/7 [00:02<00:00,  2.54it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.73it/s]
                   all         32         32      0.988          1      0.995      0.975

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      45/50      2.36G     0.2668      0.782     0.8181         16        640: 100% 7/7 [00:02<00:00,  3.43it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.15it/s]
                   all         32         32      0.988          1      0.995      0.978

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      46/50      2.36G     0.2397      0.738     0.7805         16        640: 100% 7/7 [00:03<00:00,  2.09it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.41it/s]
                   all         32         32      0.989          1      0.995      0.978

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      47/50      2.36G     0.2485      0.729     0.7804         16        640: 100% 7/7 [00:03<00:00,  2.10it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.30it/s]
                   all         32         32      0.991          1      0.995       0.98

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      48/50      2.36G     0.2454      0.733     0.7851         16        640: 100% 7/7 [00:02<00:00,  3.03it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.92it/s]
                   all         32         32      0.991          1      0.995      0.976

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      49/50      2.36G     0.2404      0.730     0.7848         16        640: 100% 7/7 [00:02<00:00,  3.33it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.93it/s]
                   all         32         32      0.991          1      0.995      0.977

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      50/50      2.36G      0.251     0.7415     0.7895         16        640: 100% 7/7 [00:03<00:00,  1.95it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  1.38it/s]
                   all         32         32      0.991          1      0.995      0.974
```

```
50 epochs completed in 0.082 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train/weights/best.pt, 5.5MB

Validating runs/detect/train/weights/best.pt...
Ultralytics 8.3.40 🚀 Python-3.10.12 torch-2.5.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLO11n summary (fused): 238 layers, 2,582,932 parameters, 0 gradients, 6.3 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 1/1 [00:00<00:00,  2.57it/s]
                   all         32         32      0.991          1      0.995       0.98
           blue-block          8          8      0.994          1      0.995      0.977
          green-block          3          3      0.979          1      0.995      0.995
            red-block          8          8      0.995          1      0.995      0.964
         yellow-block         13         13      0.995          1      0.995      0.985
Speed: 0.2ms preprocess, 2.8ms inference, 0.0ms loss, 2.4ms postprocess per image
Results saved to runs/detect/train
```

50 epochs with 160 images.

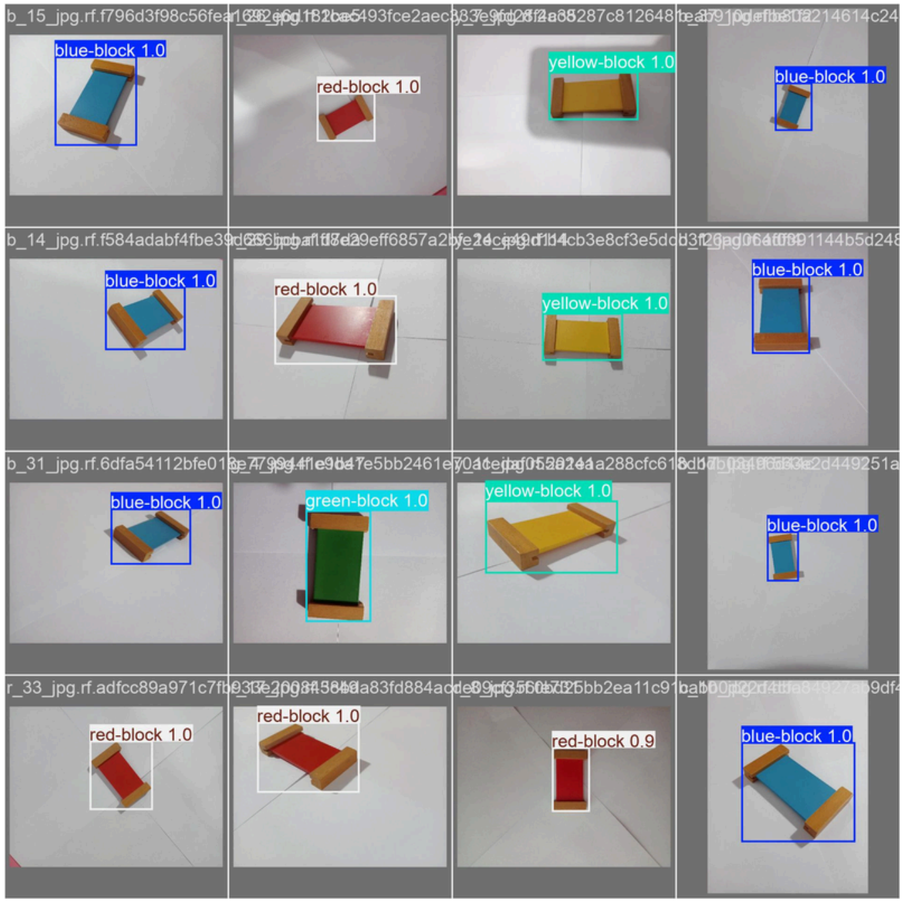
Objective 04 - Train the YOLO model to identify objects.

Evidence: Training the model with the annotated AMI Second Box of Color Pallets dataset.



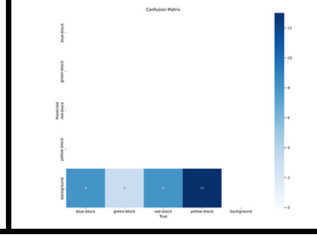
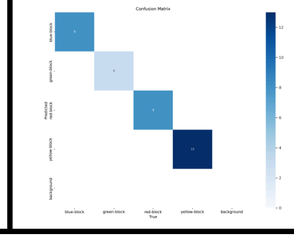
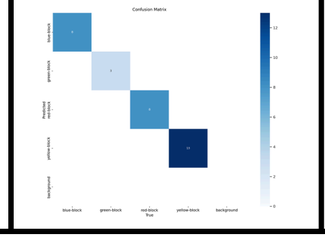
50 epochs with 160 images

The modal identified the blocks in validation dataset.



Objective 04 - Train the YOLO model to identify objects.

Comparison between Version 02 (20 Epochs) and Version 03 (50 Epochs) model training.

Metric	Version 01 5 Epochs	Version 02 20 Epochs	Version 03 50 Epochs	Comments
Precision (P)	0.00599	0.982	0.991	Version 3 shows a slightly higher overall precision.
Recall (R)	1.0	1.0	1.0	Both models achieve perfect recall (all true positives detected).
mAP@50	0.399	0.995	0.995	Model 01 has significantly lower mAP@50, which indicates it struggled to detect objects accurately at an IoU threshold of 0.5. Models 02 and 03 perform much better.
mAP@50-95	0.367	0.971	0.98	Version 3 performs better, indicating a stronger ability to generalize across IoU thresholds.
Confusion Matrix				

Objective 04 - Train the YOLO model to identify objects.

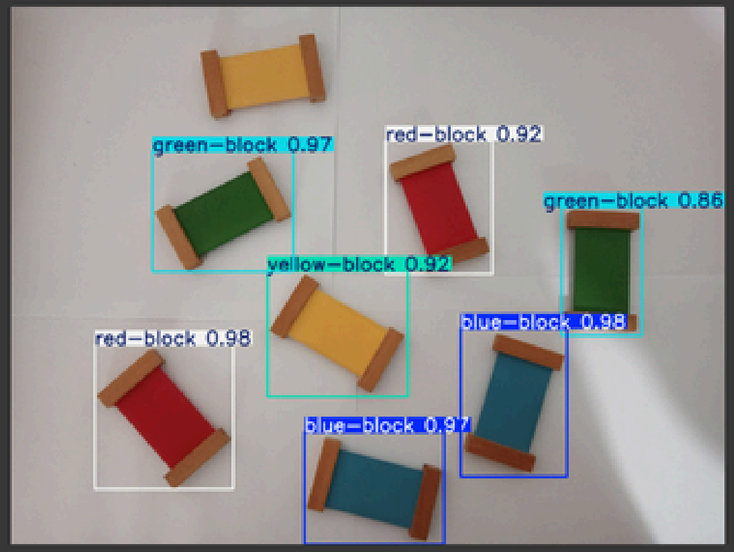
Evidence: Testing an external image with multiple color pallets

```
[ ]  1 !cp -r runs /content/drive/MyDrive/Thejani_Research_Y4/ColorPatternModel/v02-160_img_with_20_epochs
```

```
[ ]  1 # predict an external image
     2 !yolo task=detect mode=predict model={HOME}/runs/detect/train3/weights/best.pt source=/content/drive/MyDrive/Thejani_Research_Y4/ColorPatternModel/test_img02.jpg conf=0.8 save=true
```

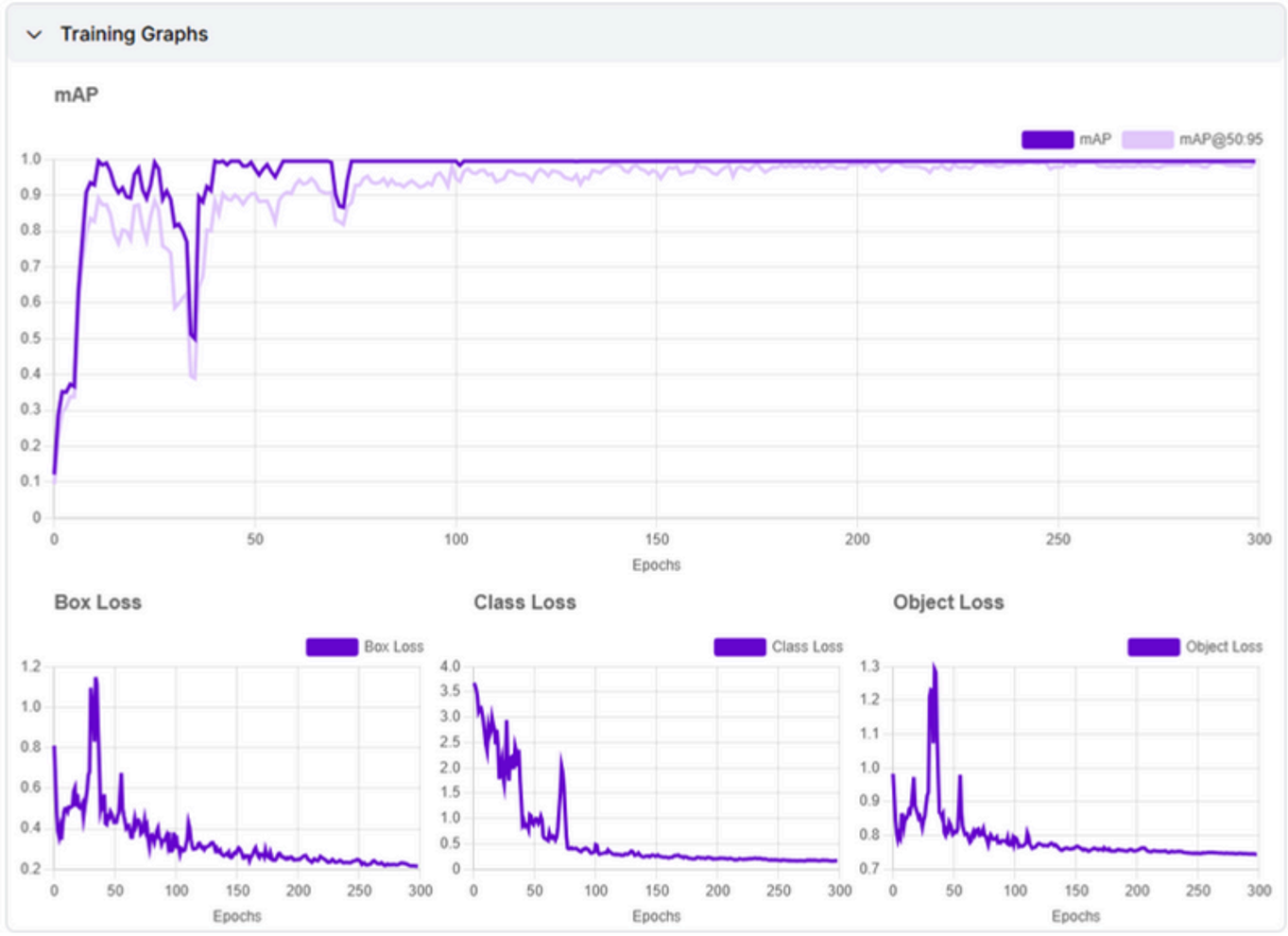
```
Ultralytics 8.3.39 🚀 Python-3.10.12 torch-2.5.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLO11n summary (fused): 238 layers, 2,582,932 parameters, 0 gradients, 6.3 GFLOPs

image 1/1 /content/drive/MyDrive/Thejani_Research_Y4/ColorPatternModel/test_img02.jpg: 512x640 2 blue-blocks, 2 green-blocks, 2 red-blocks, 1 yellow-block, 42.8ms
Speed: 4.5ms preprocess, 42.8ms inference, 565.9ms postprocess per image at shape (1, 3, 512, 640)
Results saved to runs/detect/predict6
💡 Learn more at https://docs.ultralytics.com/modes/predict
```

```
[ ]  1 IPyImage(filename=f'{HOME}/runs/detect/predict6/test_img02.jpg', width=600)
```



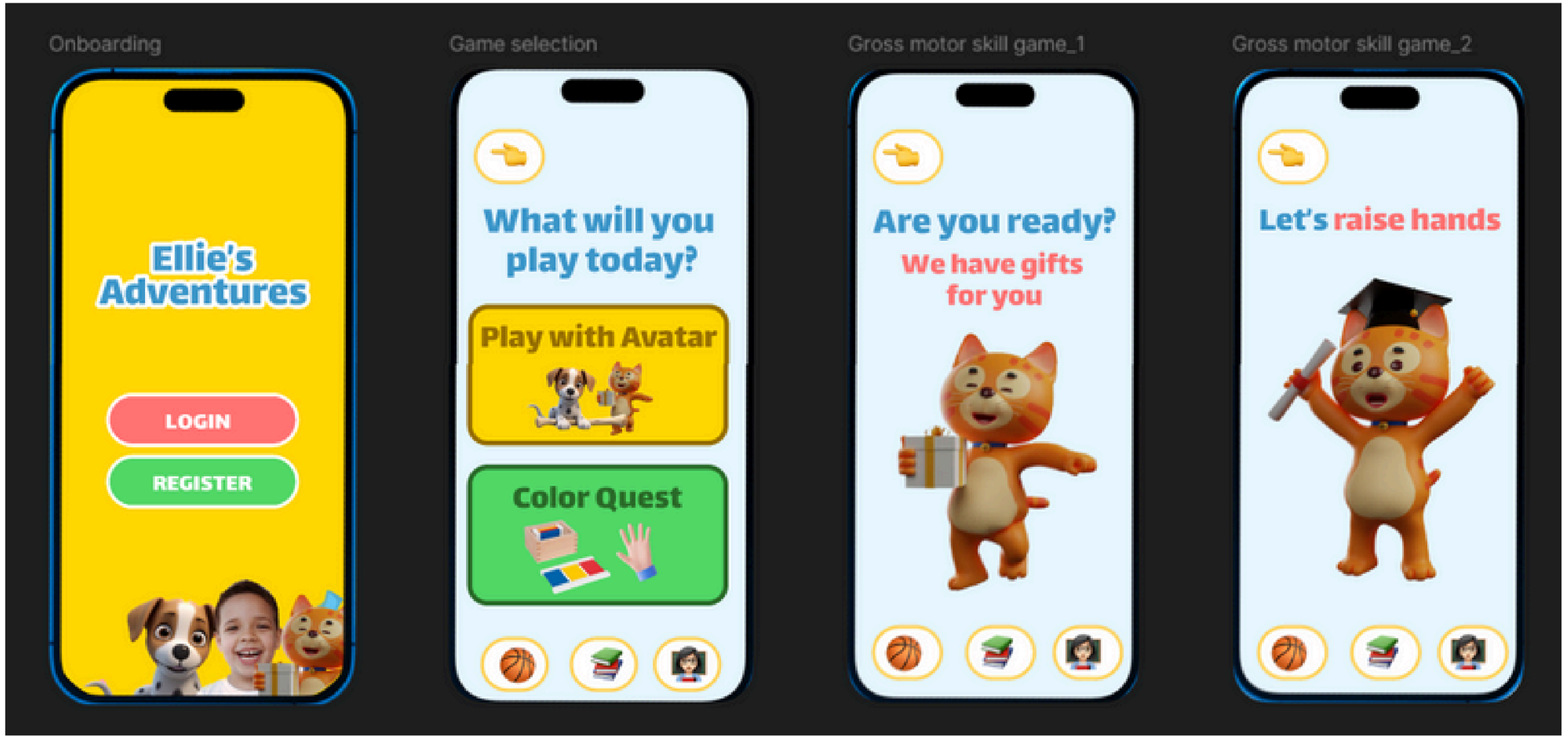
Objective 04 - Train the YOLO model to identify objects.

Evidence: Analyzed the Roboflow's trained model



Designed Figma Prototypes

These prototypes were shown to the desired user groups during field visits



Designed Figma Prototypes

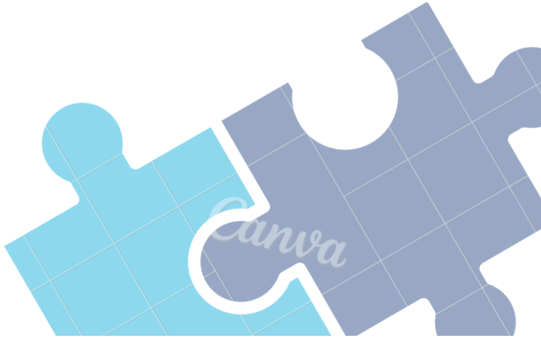
These prototypes were shown to the desired user groups during field visits

Objective Completion Status

	Status
Train the model to identify the actions performed by the child.	Completed
Identify and evaluate the child's ability to mimic the given exercises correctly.	70%
Provide feedback and encouragement based on the performance.	Pending
Train the YOLO model to identify objects correctly.	Completed
Evaluate ability of the child to create the color pattern and give the feedback.	Pending

Tools Technologies and Algorithms

Technologies	Techniques	Algorithms and Functions
Python	Keypoints extraction	LSTM
MediaPipe	Data Preprocessing	YOLO11
Tensorflow	Data annotation	
OpenCv	Object Detection	
Scikit-learn	Bound boxing	
Google Colab		
Roboflow		



Requirement Analysis

Functional requirements

- should be able to recognize activities accurately.
- should be able to provide accurate feedback.
- should be able to identify the objects correctly.
- should be able to identify and evaluate the built color pattern.
- Games should be implemented with minimal distractions to keep the child engage

Non-Functional requirements

- User-friendly Interfaces
- Quick processing action recognition system.
- Realiability by accurate activity identification and color pattern identification
- Secure, ensuring the privacy of user data.
- Availability with high functionality and minimum downtime.

Requirement Analysis

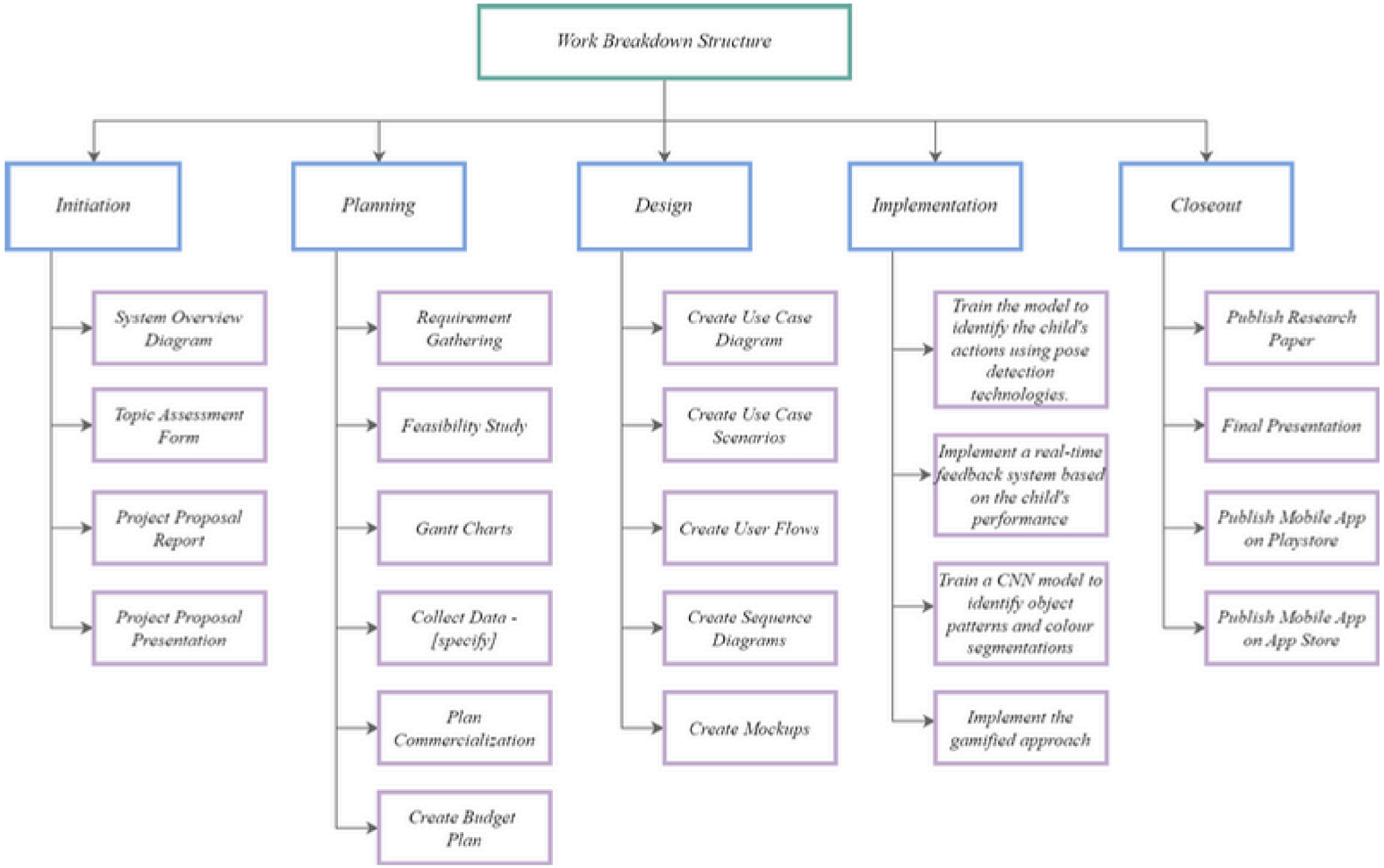
System requirements

- A device with a camera
- Adequate storage and memory to support the application
- The system should be compatible with major operating systems
- Dependencies - relevant ML libraries and game development tools.
- A strong network connection to set up and update.

Personnel requirements

- Children with autism who has speech difficulties.
- Parents and caregivers of children.
- Dr. Asiri Hewamalage

Work Breakdown Structure



References

[1] C. J. Zampella, L. A. L. Wang, M. Haley, A. G. Hutchinson and A. d. Marchena, “Motor Skill Differences in Autism Spectrum Disorder: a Clinically Focused Review,” Current psychiatry reports, 2021.

[2] A. M. Nordin, J. Ismail and N. K. Nor, “Motor Development in Children With Autism Spectrum Disorder,” Frontiers in Pediatrics, 2021.

[3] https://montessori-ami.org/

[4] https://roboflow.com/

IMPROVING FACIAL EXPRESSIONS AND EMOTIONAL SKILLS

Helapalla K. O. P. S.
IT21264016

[Software Engineering]

BACKGROUND

- Children with ASD have trouble expressing emotions as facial expressions. [1]
- Most of them have trouble recognizing the emotions of someone in front of them. [1]

LEADS TO →

- Difficulty in communication
- Social isolation
- Frustration of parents [4]

RESEARCH QUESTION

How to improve Facial expression and Emotion recognition abilities in children with ASD?

- Provide exercises to identify and associate facial expressions with emotions
- Introduce a game to identify a child's current skill in expressions
- Introduce a way to keep children involved in such exercises with rewards
- Provide exercises to practice essential facial expressions

OBJECTIVES

01 A model that detects child's FCEs and predicts the accuracy with data from camera

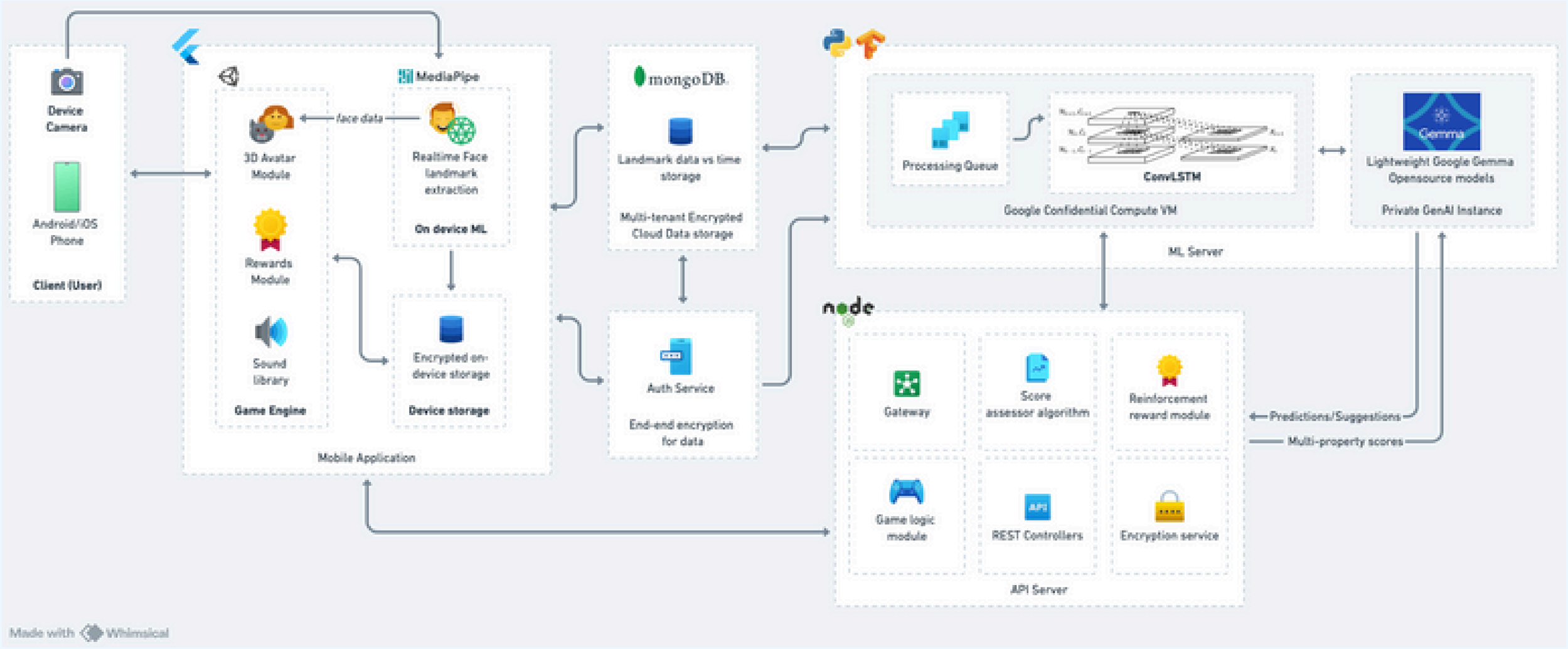
02 3D avatar that mirrors exact facial cues and blendshapes real-time from camera

03 Realtime attention tracking and constructive feedback

04 Interactive game that enhances identification skills of FCEs for the child

05 A game to determine the child's initial competencies in identifying & expressing FCEs

SYSTEM DIAGRAM



OBJECTIVE 01

A deep learning model that detects child’s FCEs and predicts the accuracy with Computer Vision

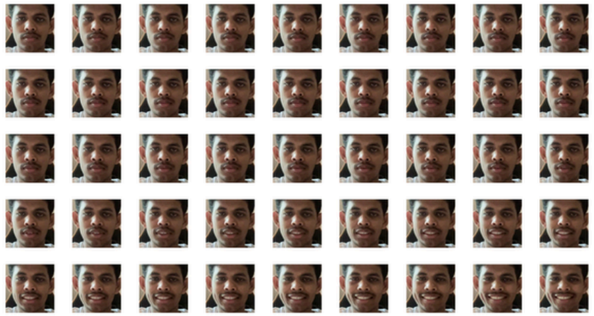
OVERVIEW

Data Preparation

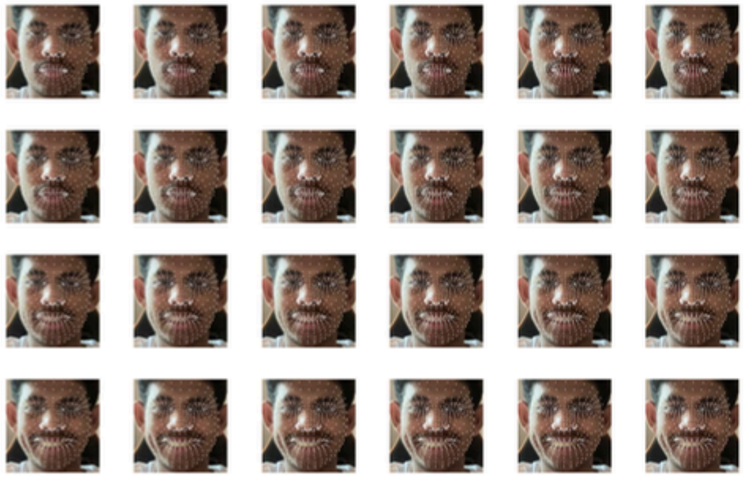
PRE-PROCESSING

135 Video Files

(Uniform sized)



Separate into frames



Apply mediapipe face landmark extraction on each frame

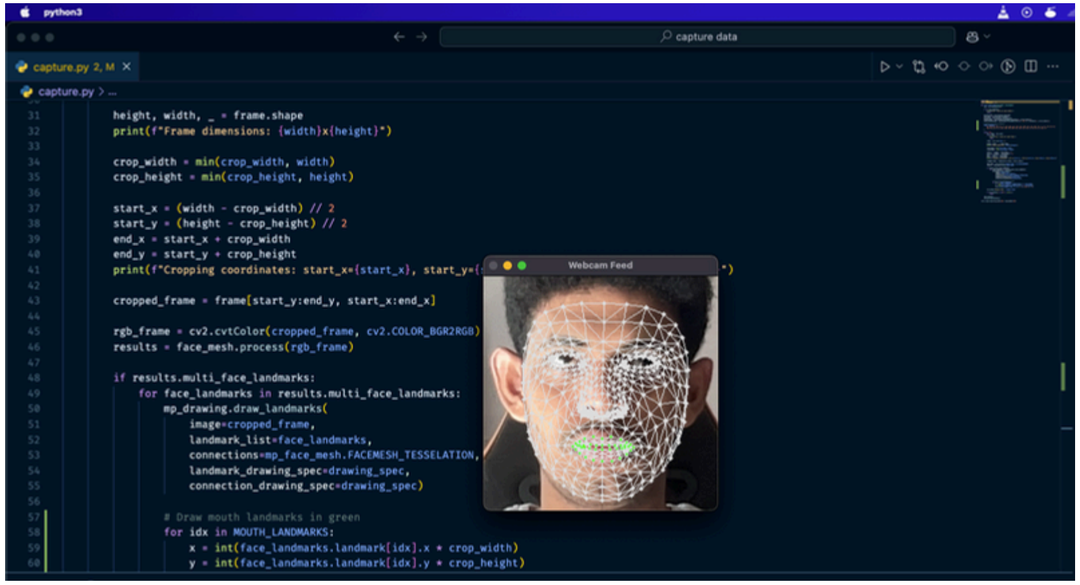
- Neutral
- Light Smile
- Full Smile

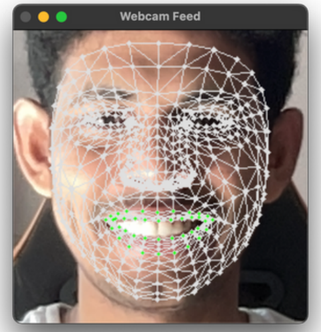
Save organized landmark data to NumPy files

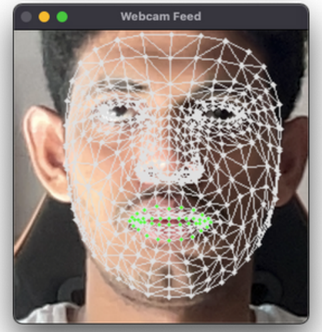
NPY

Data Capturing

Python script to capture videos of the candidate







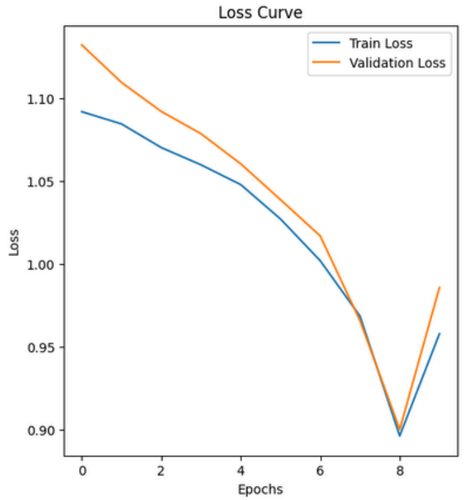
Capture the face bounding box using mediapipe landmarks and crop the background when recording

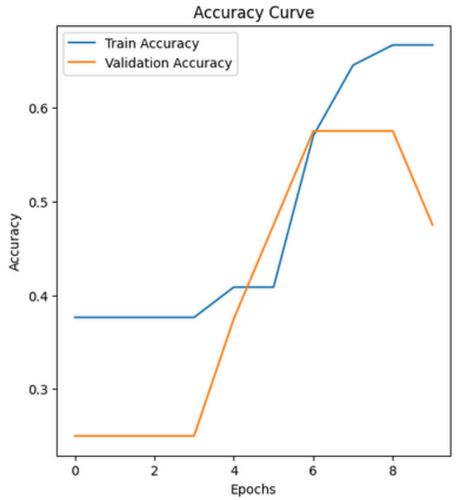
Custom Dataset

Action Intensity Classification

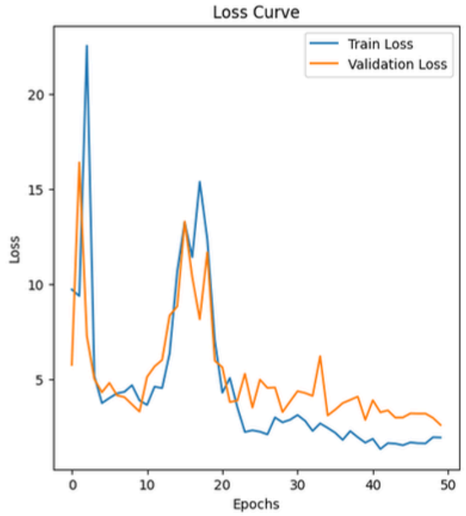
Neutral	***45 Videos 30fps, 3 Sec***
Light Smile	***45 Videos 30fps, 3 Sec***
Full Smile	***45 Videos 30fps, 3 Sec***

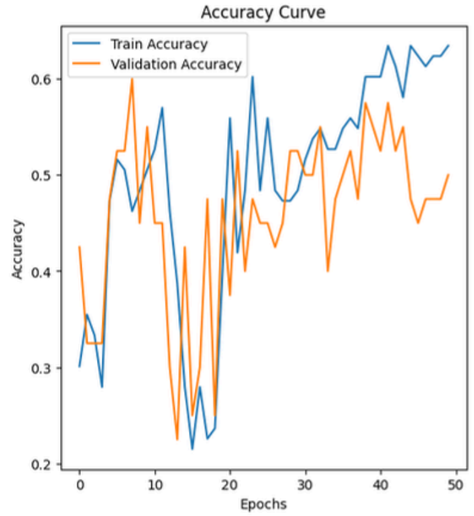
Sample size: 1
45 Videos per action





Over 10 epochs
Suggests overfitting, but can adjust and check further.





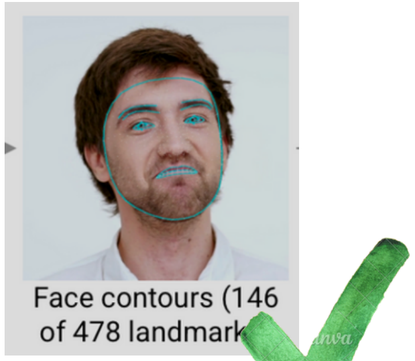
Over 50 epochs
After hyperparameter tuning, this looks decent.

Need more variant data.

Facial landmark extraction

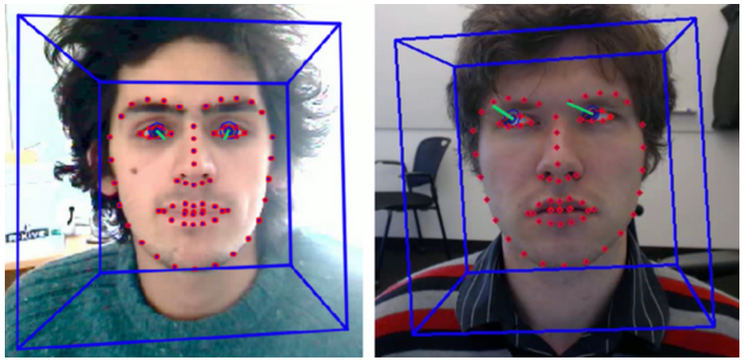
MediaPipe

- Optimized for mobile devices
- Inference time 6ms
- Provides 478 3D face landmarks [3]



OpenFace and OpenPose

- Comparatively slower
- Comparatively higher inference time
- Provides only 68 2D facial landmarks [3]



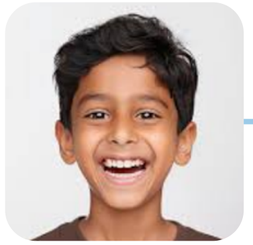
Vision Transformer for Facial Expression Recognition

motheecreator/vit-Facial-Expression-Recognition

google/vit-base-patch16-224-in21k

- ImageNet21k (14 million images)
- \+
- MMI Facial Expression Database
- \+
- FER-2013

Testing



happy: 0.97938
suprise: 0.96816

neutral: 0.95288
suprise: 0.92704
sad: 0.47202

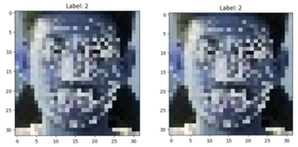
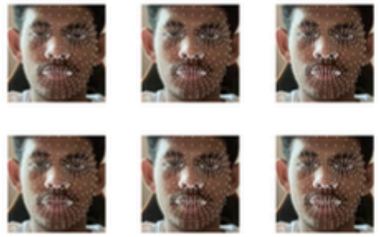
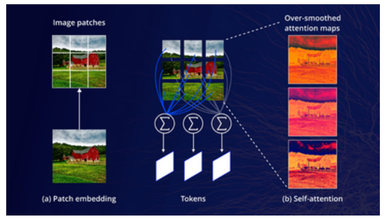
PROS

- Trained on millions of images + relevant datasets

CONS

- More resource usage for inference
- Uses images only
- No spatio-temporal feature analysis

Model Comparison

Model	Input	Training Time	Inference time	Accuracy	Media accepted
ConvLSTM + Images		High	Low	Moderate	Video (frame sequence)
LSTM + Landmark Data		Low	Low	Acceptable	Video (Landmark data (NPY arrays))
ViT (Inference)		N/A	Low	High	Images

OBJECTIVE 02

3D avatar that mirrors exact facial cues and blendshapes real-time from camera

OVERVIEW

What is ReadyPlayerMe Avatar?

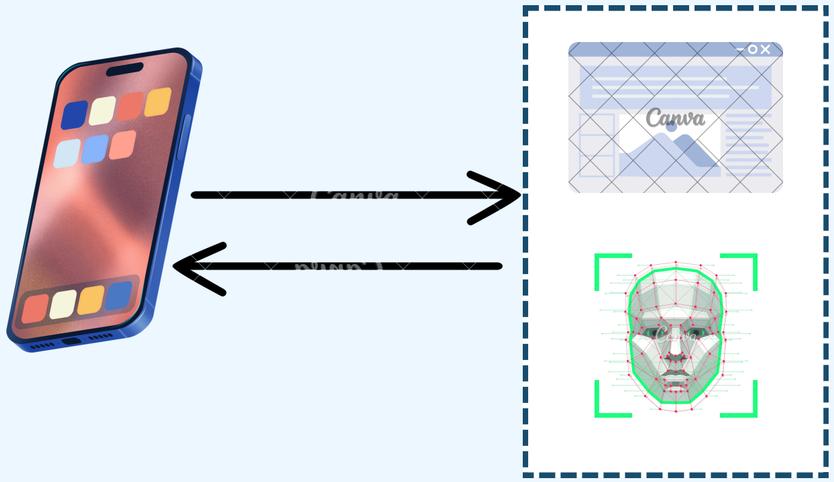
ReadyPlayerMe is a **customizeable avatar** platform that can be integrated to applications.

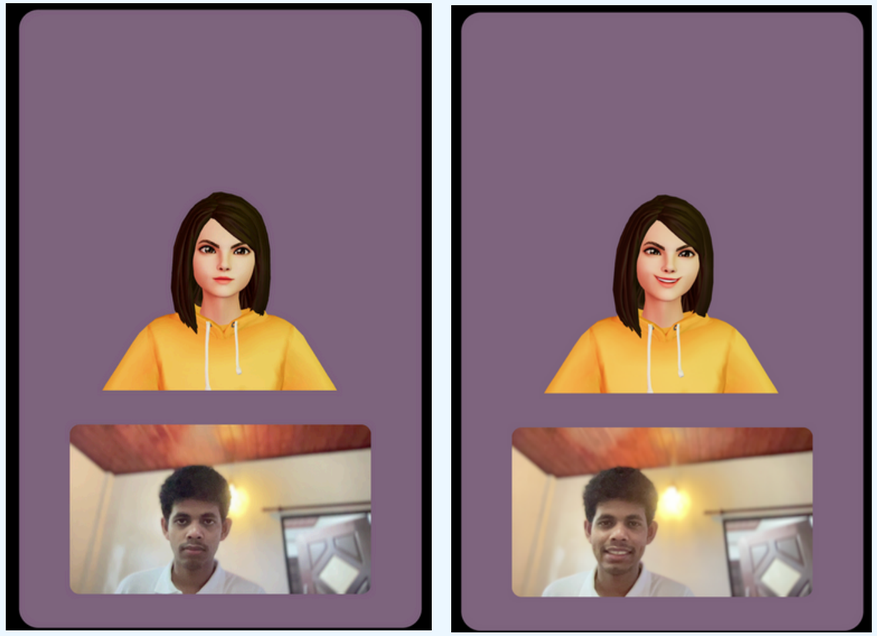
Cartoon avatar vs human avatar?

Research says that children with ASD can be more effectively trained using elements with a realistic approach. [4]

Evidence + Next Steps

- **WebView wrapper** — Since no direct react native support for Mediapipe

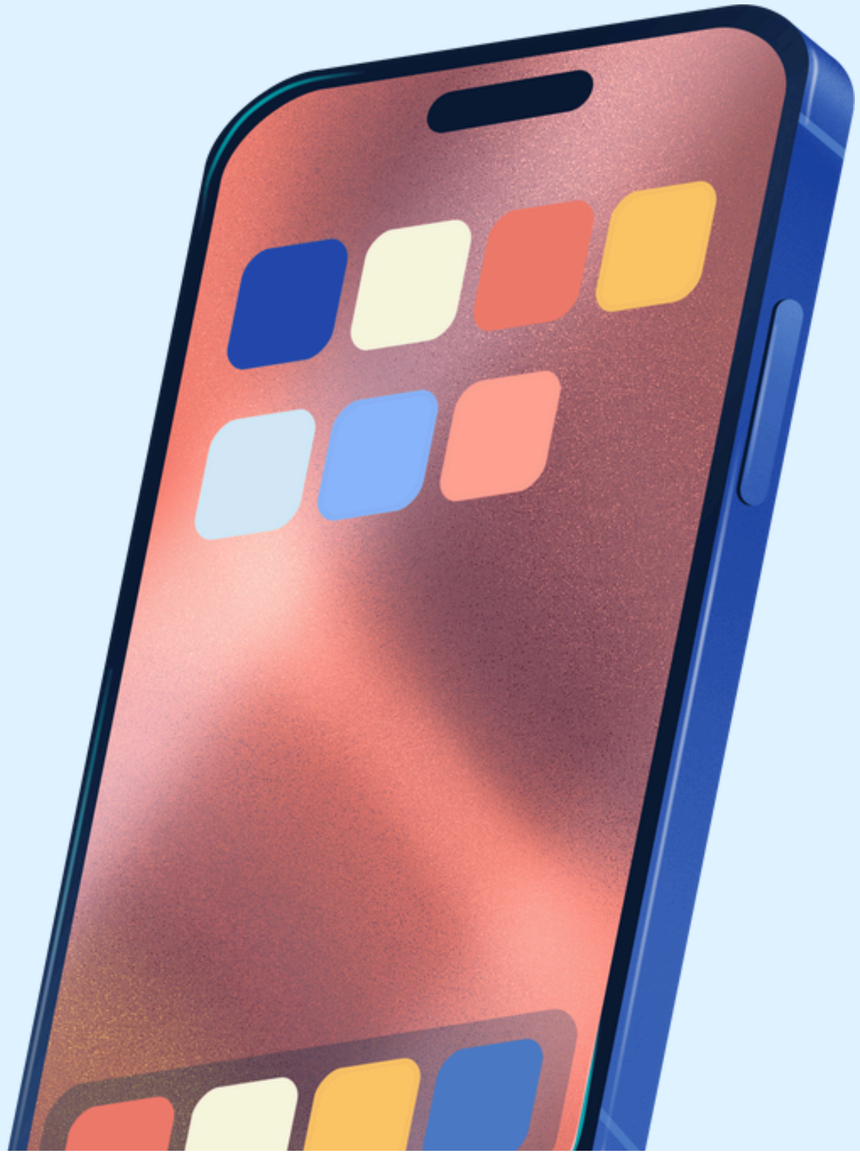




OBJECTIVE 03

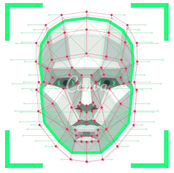
Realtime attention tracking and constructive feedback

OVERVIEW



Using OpenFace for Gaze Detection

- Provides gaze direction vector
- Provides gaze focus point



Used with a WebView wrapper

Facial landmarks around eyes

+

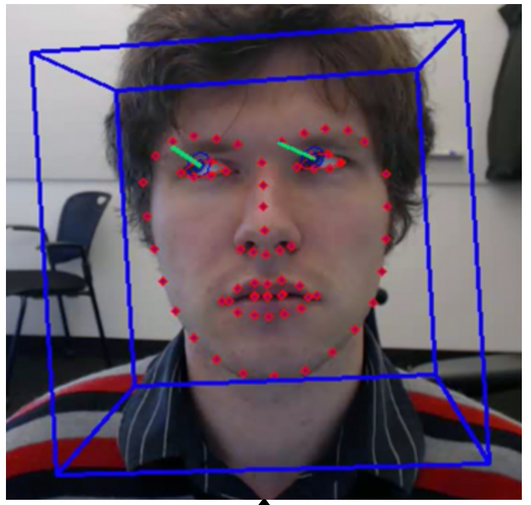
Eye Region Extraction

+

3D Head Pose Estimation

+

Gaze Direction Vector Calculation

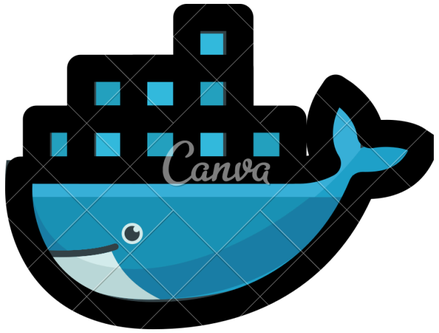


Application Screenshots

```python
from flask import Flask, jsonify
from flask_cors import CORS
import subprocess

app = Flask(__name__)
CORS(app)

@app.route('/gaze')
def gaze():
    try:
        result = subprocess.run(
            ['python', 'path/to/openface.py', '--gaze'],
            capture_output=True, text=True
        )

        gaze_direction = result.stdout.strip()

        return jsonify({'gazeDirection': gaze_direction})

    except Exception as e:
        return jsonify({'error': str(e)}), 500

if __name__ == '__main__':
    app.run(debug=True)
```

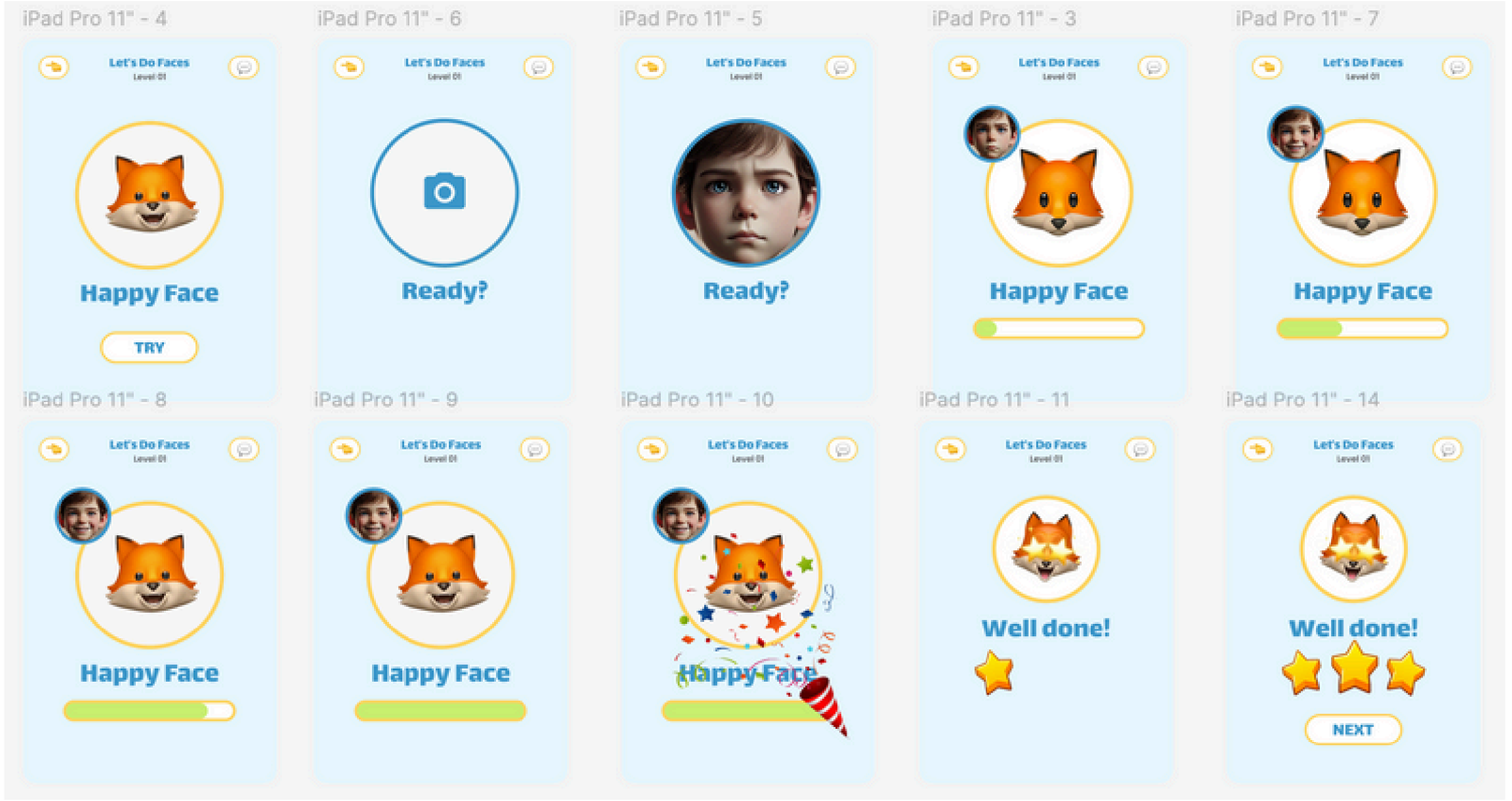
Flask Server API → `{ "gazeDirection": "away" }`

```
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do n
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 938-907-676
127.0.0.1 - - [05/Dec/2024 10:50:29] "GET /
127.0.0.1 - - [05/Dec/2024 10:50:29] "GET /
```



Prototypes


Ellie's Adventures
LOGIN
REGISTER
What will you play today?
Do faces
Match faces


iPad Pro 11" - 4
Let's Do Faces
Level 01
Happy Face
TRY
iPad Pro 11" - 6
Let's Do Faces
Level 01
Ready?
iPad Pro 11" - 5
Let's Do Faces
Level 01
Ready?
iPad Pro 11" - 3
Let's Do Faces
Level 01
Happy Face
iPad Pro 11" - 7
Let's Do Faces
Level 01
Happy Face
iPad Pro 11" - 8
Let's Do Faces
Level 01
Happy Face
iPad Pro 11" - 9
Let's Do Faces
Level 01
Happy Face
iPad Pro 11" - 10
Let's Do Faces
Level 01
Happy Face
iPad Pro 11" - 11
Let's Do Faces
Level 01
Well done!
iPad Pro 11" - 14
Let's Do Faces
Level 01
Well done!
NEXT

Tools & Technologies

Technologies

- Flutter
- Puthon
- Firestore
- Media Pipe
- TensorFlow
- Confidential
- Cloud
- VS code

Techniques

- Face Landmark detection
- Facial blendshapes
- Data Pre processing
- MES (mirroring emotion system)

Algorithms

- ConvLSTM
- Vision Transformer
- LSTM

REQUIREMENTS

Functional

- should be able to recognize speech accurately
- lit should accurately capture lip movements while pronouncing.
- should capture the words the child is finding difficult to pronounce
- Should predict new words according to the identified difficult sections.

Non-Functional

- User-friendly Interfaces
- Quick processing speech recognition system.
- Realiability by accurate speech capturing
- Secure, ensuring the privacy of user data.
- Availability with high functionality and minimum downtime.

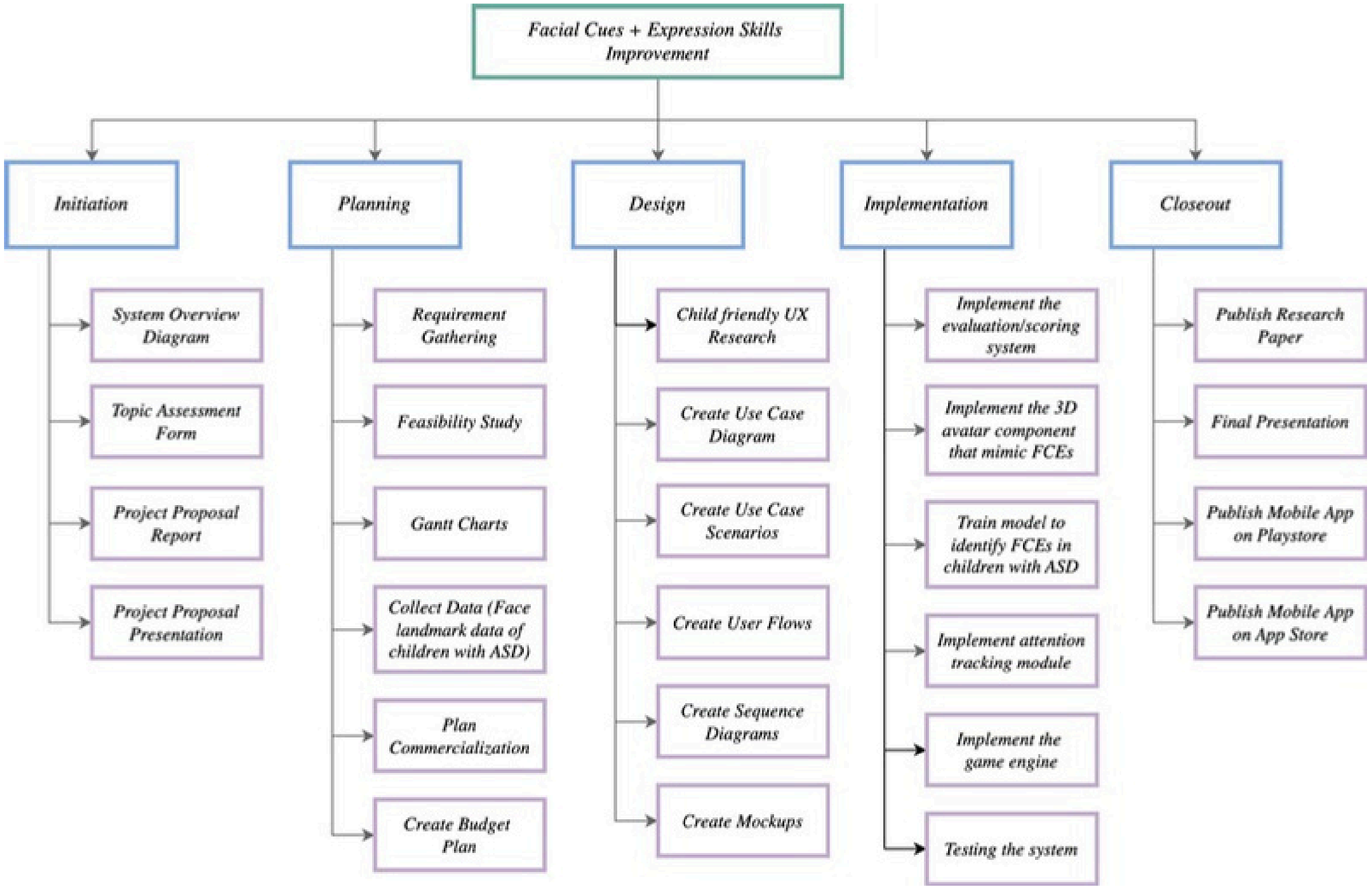
System requirements

- A device with a camera and microphone
- or Adequate storage and memory to support
- the application
- The system should be compatible with major
- operating systems
- Dependencies - relevant ML libraries and
- game development tools.

Personnel

- Children with autism who has speech difficulties.
- Parents and caregivers of children.
- Dr. Asiri Hewamalage

WBS


Facial Cues + Expression Skills Improvement
Initiation
Planning
Design
Implementation
Closeout
System Overview Diagram
Topic Assessment Form
Project Proposal Report
Project Proposal Presentation
Requirement Gathering
Feasibility Study
Gantt Charts
Collect Data (Face landmark data of children with ASD)
Plan Commercialization
Create Budget Plan
Child friendly UX Research
Create Use Case Diagram
Create Use Case Scenarios
Create User Flows
Create Sequence Diagrams
Create Mockups
Implement the evaluation/scoring system
Implement the 3D avatar component that mimic FCEs
Train model to identify FCEs in children with ASD
Implement attention tracking module
Implement the game engine
Testing the system
Publish Research Paper
Final Presentation
Publish Mobile App on Playstore
Publish Mobile App on App Store

OBJECTIVES

01 A model that detects child's FCEs and predicts the accuracy with data from camera — 100%

02 3D avatar that mirrors exact facial cues and blendshapes real-time from camera — 100%

03 Realtime attention tracking and constructive feedback — 50%

04 Interactive game that enhances identification skills of FCEs for the child

05 A game to determine the child's initial competencies in identifying & expressing FCEs

REFERENCES

1. C. Tsangouri, W. Li, Z. Zhu, F. Abtahi and T. Ro, "An interactive facial-expression training platform for individuals with autism spectrum disorder," 2016 IEEE MIT Undergraduate Research Technology Conference (URTC), Cambridge, MA, USA, 2016, pp. 1-3, doi: 10.1109/URTC.2016.8284067
2. S. Jain, B. Tamersoy, Y. Zhang, J. K. Aggarwal and V. Orvalho, "An interactive game for teaching facial expressions to children with Autism Spectrum Disorders," 2012 5th International Symposium on Communications, Control and Signal Processing, Rome, Italy, 2012, pp. 1-4, doi: 10.1109/ISCCSP.2012.6217849
3. A. S. Shminan, L. J. Choi and S. Sharif, "AutiTEACCH: Mobile-based Learning in a Structured Teaching Approach for Autistic Children Caregivers," 2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 2020, pp. 259-264, doi: 10.1109/ICIMCIS51567.2020.9354288
4. P. Leijdekkers, V. Gay and F. Wong, "CaptureMyEmotion: A mobile app to improve emotion learning for autistic children using sensors," Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems, Porto, Portugal, 2013, pp. 381-384, doi: 10.1109/CBMS.2013.6627821.

IMPROVING COGNITIVE & VERBAL COMMUNICATION SKILLS

Jathurshan Manistar
IT21246296

Information Technology

BACKGROUND

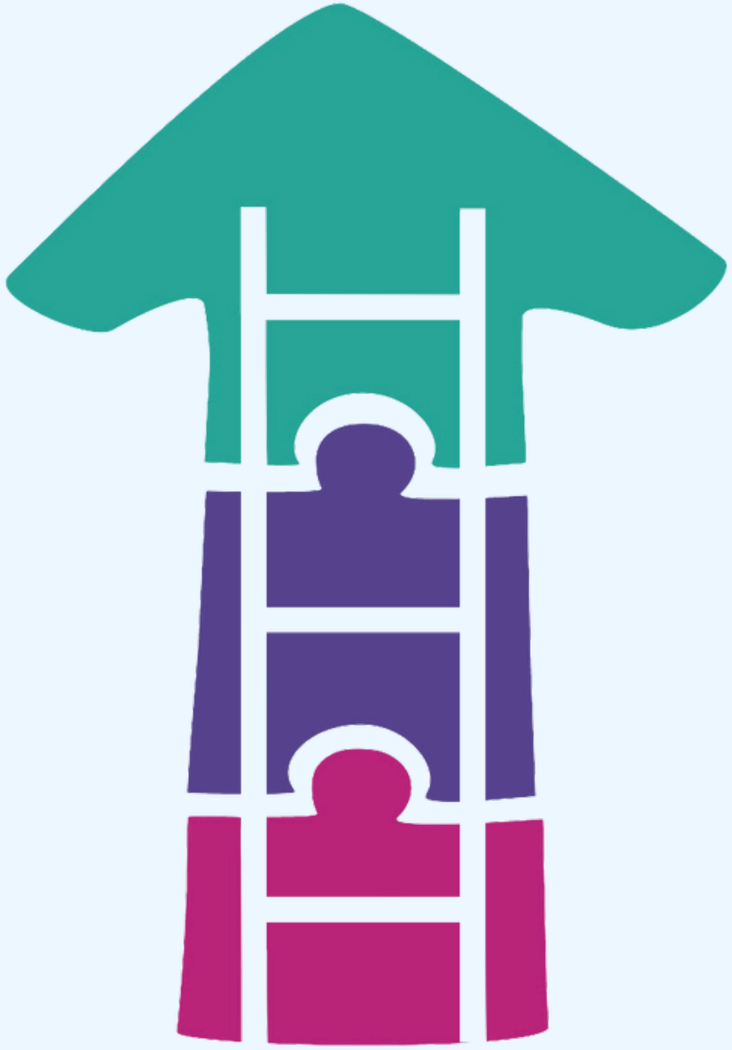
- Children with ASD have **difficulties with communication**, leading to isolation and frustration.
- These challenges **impact daily interactions** and overall well-being.
- Enhancing these skills is crucial for **better social interactions** and **quality of life**.
- Enabling the child to **navigate daily life more effectively**.

MAIN AND SPECIFIC OBJECTIVES

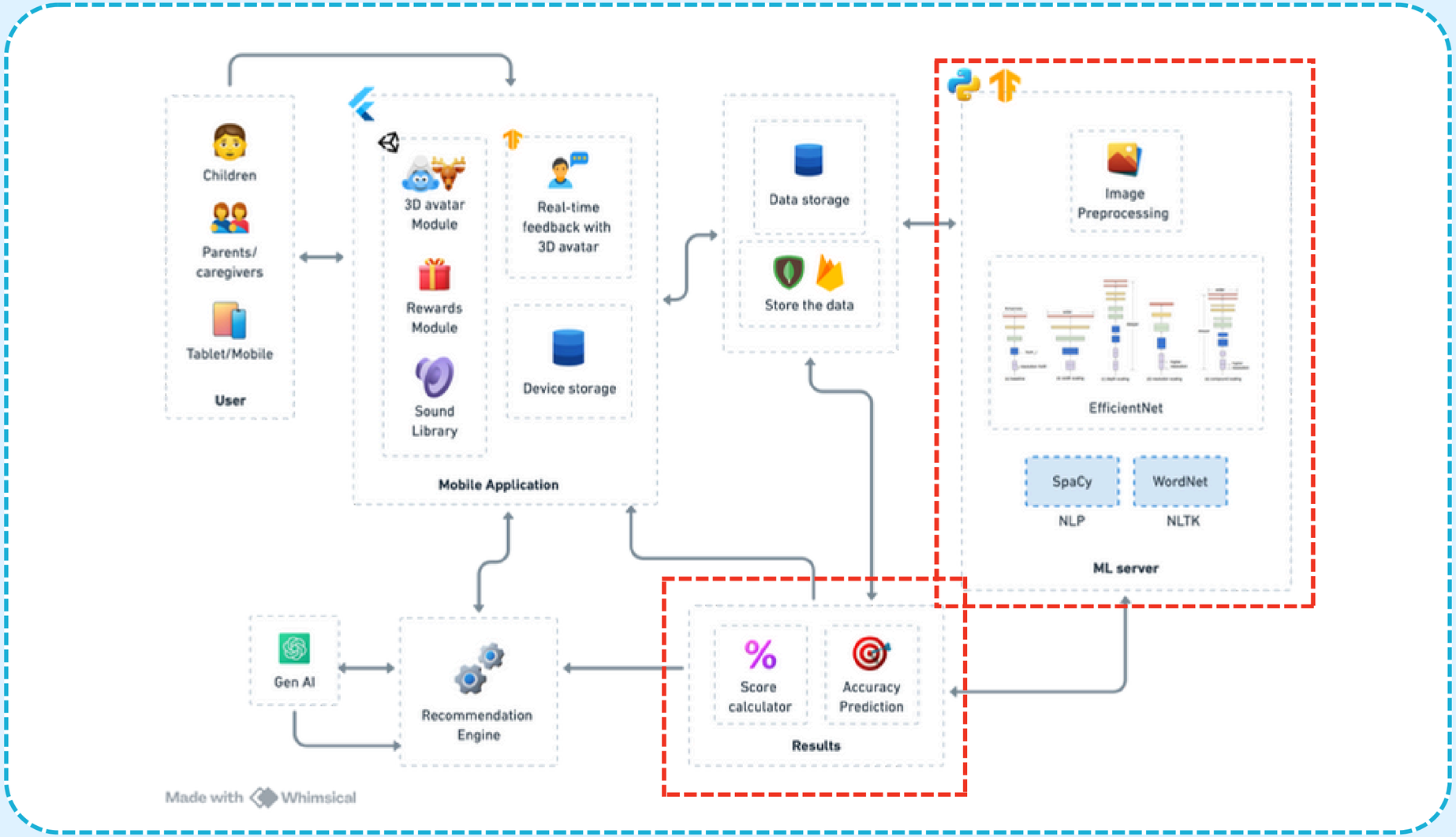
IMPROVEMENT OF COGNITIVE & VERBAL COMMUNICATION SKILLS THROUGH A SERIES OF GAMES AND EXERCISES.

- Identify the child's current cognitive and verbal communication skills.
- Analyze written letters from children to identify the current context.
- Predict the accuracy of hand-drawn letters for verbal assessment.
- Developing a game to building shapes from blocks to enhance cognitive skills.
- Employ DL models and rule-based systems for word prediction, accuracy of the letter and provide feedback.

METHODOLOGY AND EVIDENCE OF COMPLETION



SYSTEM DIAGRAM



OBJECTIVE 01:

Identify the child's current cognitive and verbal communication skills.

Data collection and Preprocessing

[1]

Question
What is your gender: boy or girl?
What is the name of your neighborhood?
What is your age?
1. What avatar do you look more like?
2. Did you like today's avatar?
3. Would you have enjoyed the same with the other [mestizo/Afro-Ecuadorian/white] avatar?
4. Which [boy/girl] avatar would you choose the second? Would you have enjoyed the same with [him/her]?
5. Did you like the way the avatar greeted you?
6. Did you like the clothes the avatar wore?
7. Did you like talking with the avatar?
8. Did you understand what the avatar said?
9. Did you like the avatar's voice?
10. Did you like the way the avatar moved?
11. Did the avatar scare you at some point?
12. Would you like to talk to the avatar again?
13. Would you like to have the avatar at school?
14. Would you like to have the avatar at home?
15. Do you think the avatar looks like your classmates and/or relatives?
16. Did you like the way the avatar said goodbye?
17. Did you like the mood of the avatar?

[2]

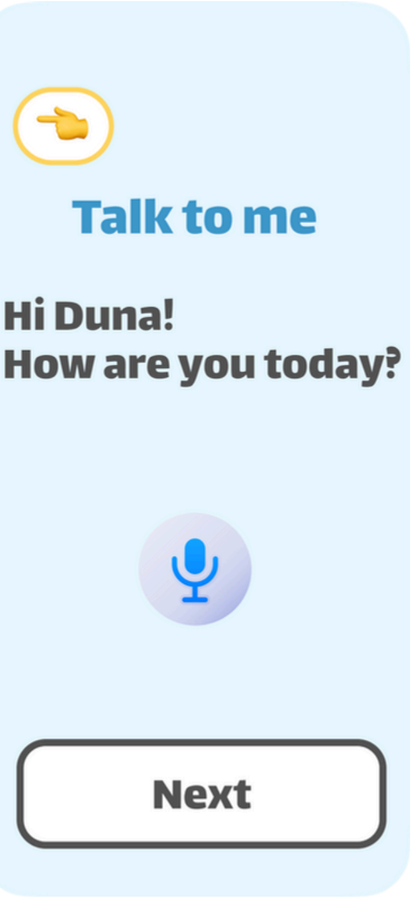
A. Session 01: Introduction

During the first session, we are trying to introduce NAO robot with children and their caregivers. Some parents also get involved to learn our working principles.

B. Session 02: Questions & Answering with NAO Robot

The session starts with the robot saying 'Hello' to the Children's. NAO also repeated 'Hello' word together with 'Good Morning'. In this session NAO talked with each of the Children's separately. NAO asked the same question listed view bellow to each child. There was a break in between two questions which is around 20 seconds.

a) How are you?
b) What is your name?
c) What is your father's name?
d) What is your mother's name?
e) Where do you live?
f) How old are you?
g) Are you happy?
h) Do you like going to school?
i) Do you like Chocolates?
j) What is this Robot's Name?



OBJECTIVE 01:

Identify the child's current cognitive and verbal communication skills.

Scoring system and Validation method

[2]

Social Communication Questionnaire (SCQ): a Screening Measure for ASD

The SCQ is a brief, 40-item, parent-report screening measure that focuses on items relating to ASD symptomatology likely to be observed by a primary caregiver. Although the SCQ is a screening tool—and, thus, cannot be used for diagnosis of ASD—it is based on the Autism Diagnostic Interview (ADI-R) [17], a semi-structured parent interview conducted by a trained clinician or researcher that can be used for diagnostic evaluation of children with suspected ASD.

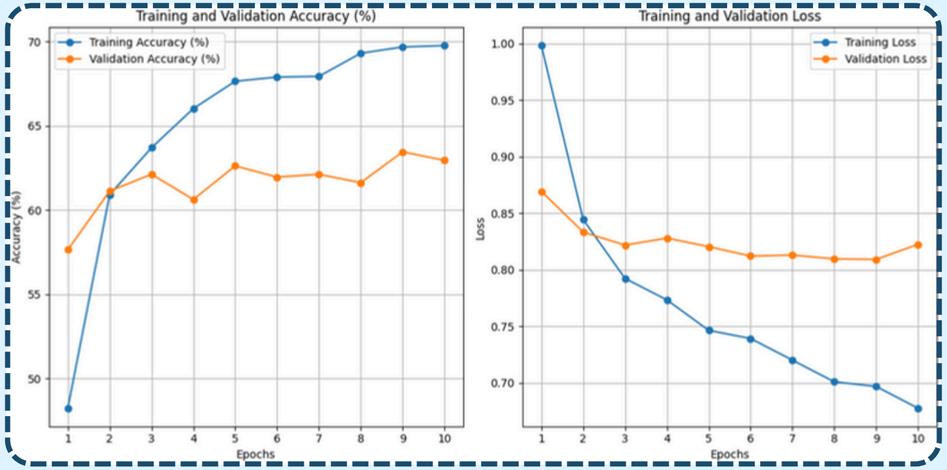
Each item in the SCQ requires a dichotomous "yes"/"no" response, and each scored item receives a value of 1 point for abnormal behavior and 0 points for absence of abnormal behavior/normal behavior. The first item—"Is she/he now able to talk using short phrases or sentences?"—is not scored, but rather determines whether six items relating to abnormal language are assigned. Only "verbal" children (i.e., children

```python
# Define validation functions
def contains_idiom(sentence):
    """Checks if the sentence contains any common idioms."""
    for idiom in common_idioms:
        if idiom in sentence:
            return True
    return False

def validate_label(response, label):
    """Validates the sentence against its label."""
    words = word_tokenize(response)
    word_count = len(words)
    sentence_complexity = word_count > 12  # Example: More than 12 words is considered complex

    if label == 0 and (sentence_complexity or contains_idiom(response)):  # Label 0 corresponds to 'beginner'
        print(f"Warning: The 'beginner' label might be incorrect for this sentence: {response}")
    elif label == 1 and contains_idiom(response):  # Label 1 corresponds to 'intermediate'
        print(f"Warning: The 'intermediate' label might be incorrect for this sentence: {response}")
    elif label == 2 and not contains_idiom(response):  # Label 2 corresponds to 'advanced'
        print(f"Warning: The 'advanced' label might be incorrect for this sentence: {response}")
```

OBJECTIVE 01:

Identify the child's current cognitive and verbal communication skills.

Approach 1: Training the Model Using a CNN-Based



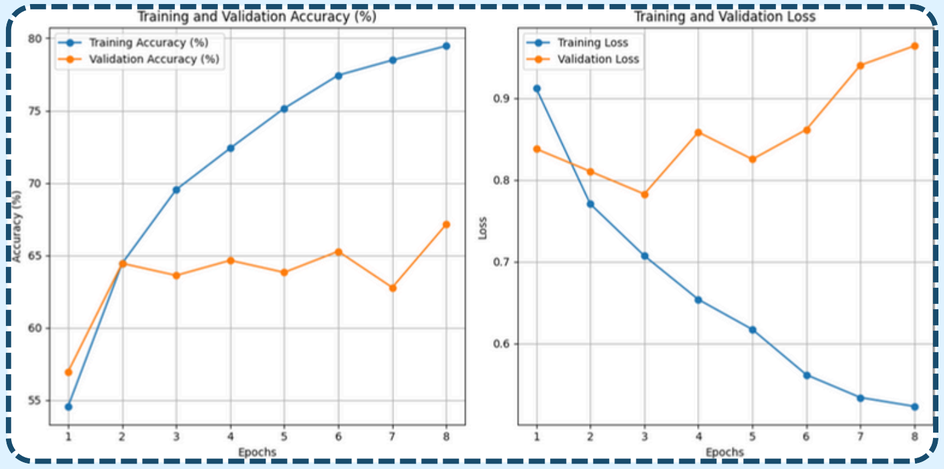

```python
# simple neural network
model = Sequential()

model.add(Dense(64, activation='relu', input_shape=(X_train.shape[1],))) #Input layer with Dense and Dropout
model.add(Dropout(0.3))

model.add(Dense(32, activation='relu')) #Hidden layer
model.add(Dropout(0.3))

model.add(Dense(3, activation='softmax'))  #Output layer #3 categories: Beginner, Intermediate, Advanced

# Compile the model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Train the model with 10 epochs
model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test), batch_size=4)

loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {accuracy * 100:.2f}%")
```

```
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9823 - loss: 0.0574 - val_accuracy: 0.4921 - val_loss: 2.5116
Epoch 48/50
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9610 - loss: 0.0659 - val_accuracy: 0.4921 - val_loss: 2.5444
Epoch 49/50
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9686 - loss: 0.0715 - val_accuracy: 0.4762 - val_loss: 2.6021
Epoch 50/50
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9608 - loss: 0.0798 - val_accuracy: 0.4921 - val_loss: 2.5655
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 393ms/step - accuracy: 0.5155 - loss: 2.3121
Test Accuracy: 49.21%
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 107ms/step
Predicted Category for 'I played with my dog today and had a lot of fun.': Beginner
```

OBJECTIVE 01:

Identify the child's current cognitive and verbal communication skills.

Approach 2: Training the Model Using a RNN-Based Approach - [Best model]




```python
model = Sequential() # Define the model

model.add(Embedding(input_dim=10000, output_dim=128, input_length=100)) # Embedding layer
model.add(Bidirectional(LSTM(64, return_sequences=True))) # First Bidirectional LSTM layer with Dropout
model.add(Dropout(0.3))
model.add(Bidirectional(LSTM(64, return_sequences=True))) # Second Bidirectional LSTM layer with Dropout
model.add(Dropout(0.3))
model.add(GRU(64)) # GRU layer with Dropout
model.add(Dropout(0.3))
model.add(Dense(32, activation='relu')) # Dense layer with Dropout
model.add(Dropout(0.3))
model.add(Dense(3, activation='softmax')) # Output layer

model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

```
241/241 ━━━━━━━━━━━━━━━━━━━━ 6s 27ms/step - accuracy: 0.8780 - loss: 0.2787 - val_accuracy: 0.6715 - val_loss: 1.9395
Epoch 48/50
241/241 ━━━━━━━━━━━━━━━━━━━━ 6s 23ms/step - accuracy: 0.8679 - loss: 0.2548 - val_accuracy: 0.6445 - val_loss: 2.0068
Epoch 49/50
241/241 ━━━━━━━━━━━━━━━━━━━━ 10s 23ms/step - accuracy: 0.8808 - loss: 0.2625 - val_accuracy: 0.6590 - val_loss: 2.0720
Epoch 50/50
241/241 ━━━━━━━━━━━━━━━━━━━━ 11s 27ms/step - accuracy: 0.8837 - loss: 0.2356 - val_accuracy: 0.6611 - val_loss: 2.1080
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 0.6282 - loss: 2.0698
Test Accuracy: 67.29%
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 322ms/step
Predicted Category for 'I played with my dog today and had a lot of fun.': Intermediate
```

OBJECTIVE 01:

Identify the child's current cognitive and verbal communication skills.

Test cases with the approach 2 - [Best model]

```
# List of sentences to predict
testcases = [
    "cat is",
    "The quick brown fox jumps over the lazy dog",
    "He’s in hot water after missing the deadline"
]

# Tokenize and pad all the sentences at once
new_sequences = tokenizer.texts_to_sequences(testcases)
new_padded_sequences = pad_sequences(new_sequences, maxlen=100)

# Make predictions for all sentences at once
predicted_labels = model.predict(new_padded_sequences)

# Inverse transform to get the category labels
predicted_categories = label_encoder.inverse_transform(np.argmax(predicted_labels, axis=1))

# Print predictions for each sentence
for sentence, category in zip(testcases, predicted_categories):
    print(f"Predicted Category for '{sentence}': {category}")
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step
Predicted Category for 'cat is': Beginner
Predicted Category for 'The quick brown fox jumps over the lazy dog': Advanced
Predicted Category for 'He’s in hot water after missing the deadline': Intermediate
```

Test Case	Identify the current cognitive level
Test Scenario	Analyze the sentence and predict the level

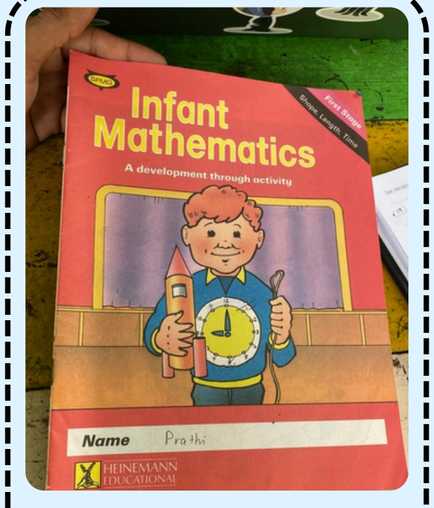
Input	“cat is”
Expected Output	Beginner
Actual output	Beginner
Status (Pass/ Fail)	Pass
Input	"The quick brown fox jumps over the lazy dog"
Expected Output	Intermediate
Actual output	Intermediate
Status (Pass/ Fail)	Pass
Input	"He’s in hot water after missing the deadline"
Expected Output	Advanced
Actual output	Advanced
Status (Pass/ Fail)	Pass

OBJECTIVE 02:

Analyze written letters from children to identify the current context.

Data collection and Preprocessing

Idea

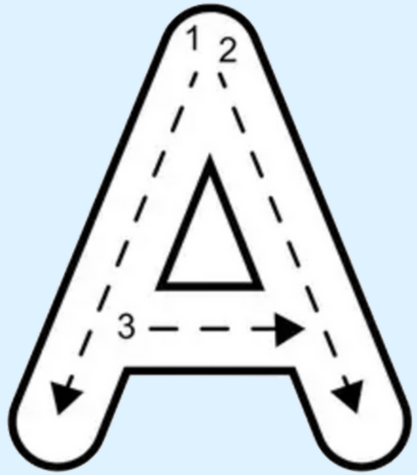


The Handbook of Assessment is used in homes of children with ASD.

Preprocessing

```python
# Function to preprocess the image
def preprocess_image(img_path):
    img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)  # Load as grayscale
    img = cv2.resize(img, (28, 28))  # Resize to 28x28
    img = img.astype('float32') / 255  # Normalize to range [0, 1]
    img = np.expand_dims(img, axis=-1)  # Add channel dimension
    return np.expand_dims(img, axis=0)  # Add batch dimension

# Path to your test image
image_path = '/content/A.jpg'  # Replace with actual image path
preprocessed_image = preprocess_image(image_path)

# Make a prediction
prediction = model.predict(preprocessed_image)
predicted_class = np.argmax(prediction)
print(f"Predicted class: {predicted_class} (Corresponding to letter: {chr(predicted_class + 65)})")

# Display the image and prediction
plt.imshow(preprocessed_image[0].reshape(28, 28), cmap='gray')
plt.title(f'Predicted Class: {chr(predicted_class + 65)}')
plt.show()
```

Assessment

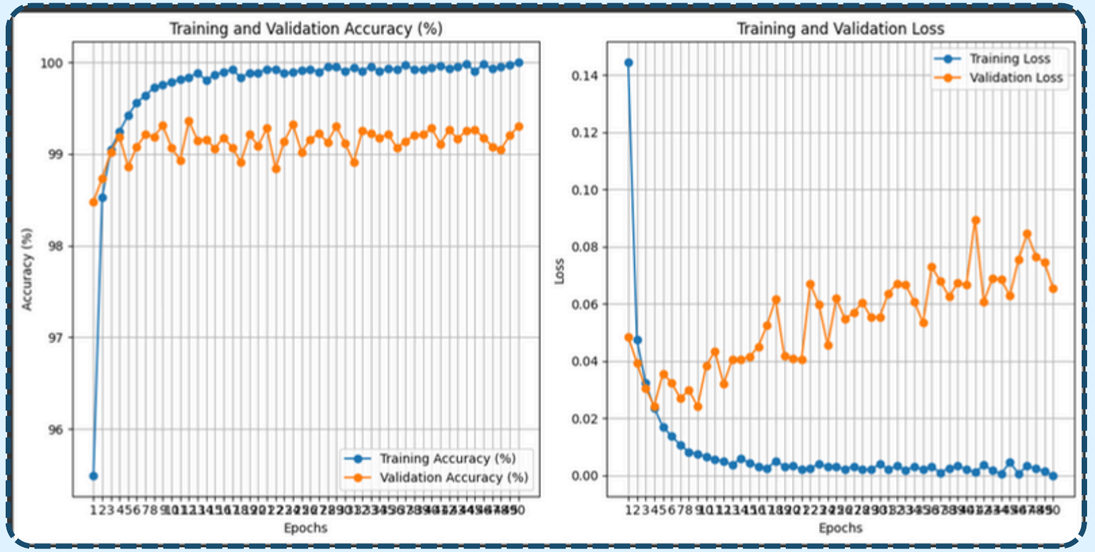


OBJECTIVE 02:

Analyze written letters and sentence structures from children to identify the current context.

Training the Model Using a CNN Approach

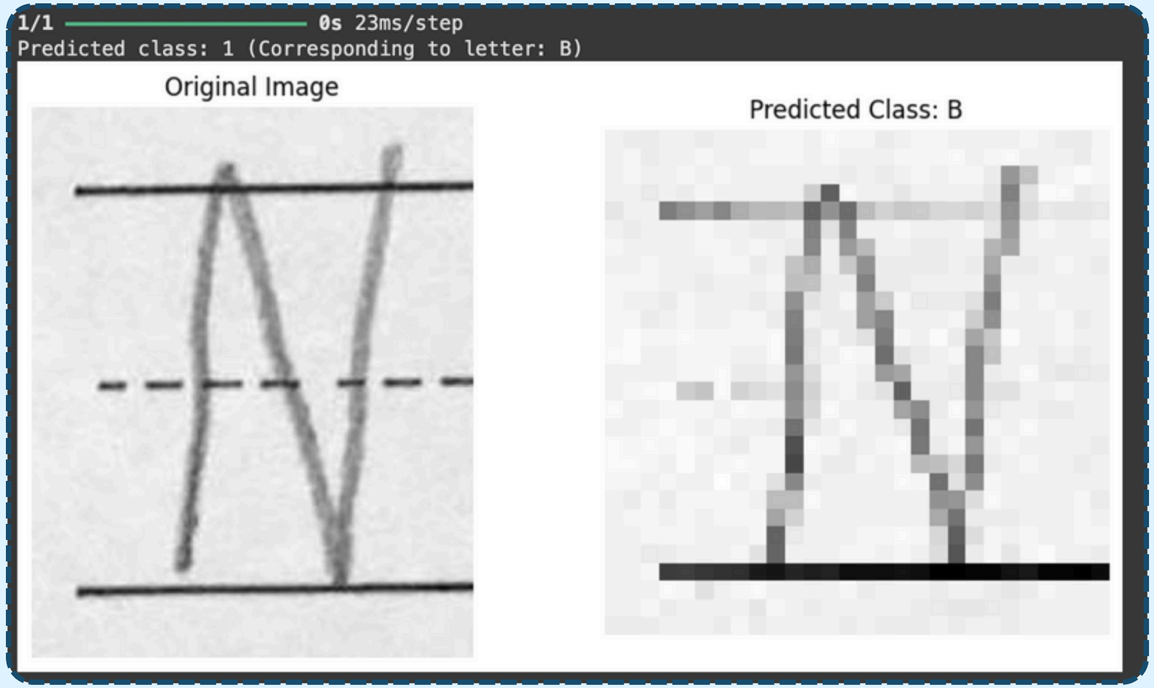
```python
# Placeholder: Using MNIST for digit recognition
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Preprocess data
X_train = X_train.reshape(X_train.shape[0], 28, 28, 1).astype('float32') / 255
X_test = X_test.reshape(X_test.shape[0], 28, 28, 1).astype('float32') / 255

# Convert labels to categorical (one-hot encoding)
y_train = to_categorical(y_train, 26)  # For alphabets A-Z
y_test = to_categorical(y_test, 26)

# Model creation
model = Sequential()
# Convolutional layer with 32 filters, a 3x3 kernel, and ReLU activation
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPooling2D((2, 2))) # Max pooling layer to reduce spatial dimensions (2x2 pool size)
model.add(Conv2D(64, (3, 3), activation='relu')) #2nd convolutional layer with 64 filters and ReLU activation
model.add(MaxPooling2D((2, 2))) # 2nd max pooling layer to further reduce spatial dimensions
model.add(Flatten()) # Flatten the 2D output to 1D for the fully connected layer
model.add(Dense(128, activation='relu')) # Fully connected (dense) layer with 128 neurons and ReLU activation
model.add(Dense(26, activation='softmax'))  # 26 output classes for A-Z

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))
```



OBJECTIVE 03:

Predict the accuracy of hand-drawn letters for verbal assessment.

Testcases

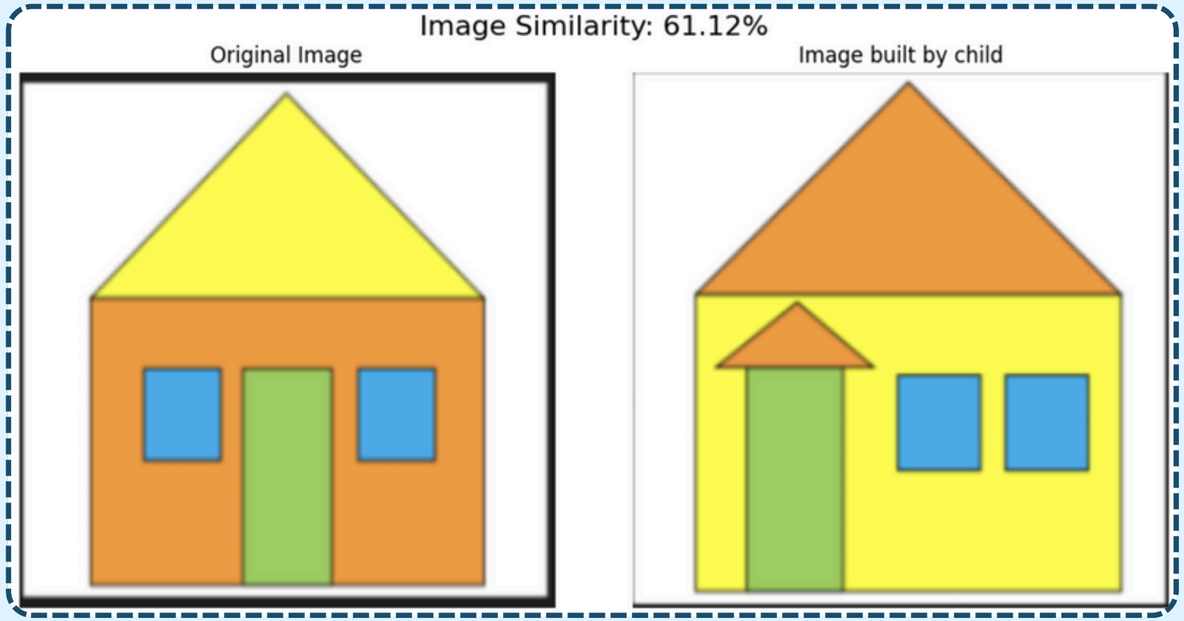


OBJECTIVE 04:

Developing a game to building shapes from blocks to enhance cognitive skills.

Find the similarity between two images

```python
import cv2
from skimage.metrics import structural_similarity as compare_ssim

def compare_images(image1_path, image2_path):
    # Load the images
    img1 = cv2.imread(image1_path, cv2.IMREAD_COLOR)
    img2 = cv2.imread(image2_path, cv2.IMREAD_COLOR)

    # Resize images to the same dimensions (adjust size as needed for puzzles)
    img1 = cv2.resize(img1, (300, 300))
    img2 = cv2.resize(img2, (300, 300))

    # Convert images to grayscale
    gray1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
    gray2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)

    # Compute Structural Similarity Index (SSIM)
    score, _ = compare_ssim(gray1, gray2, full=True)

    # Convert SSIM score to percentage
    percentage_similarity = score * 100

    return percentage_similarity
```

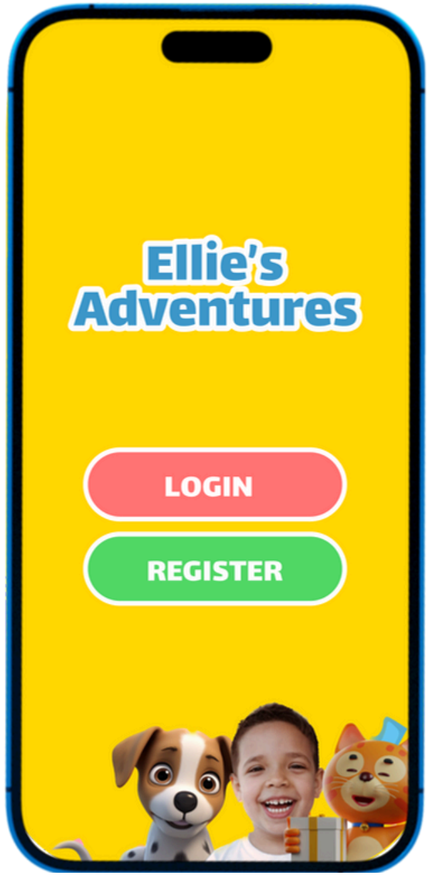


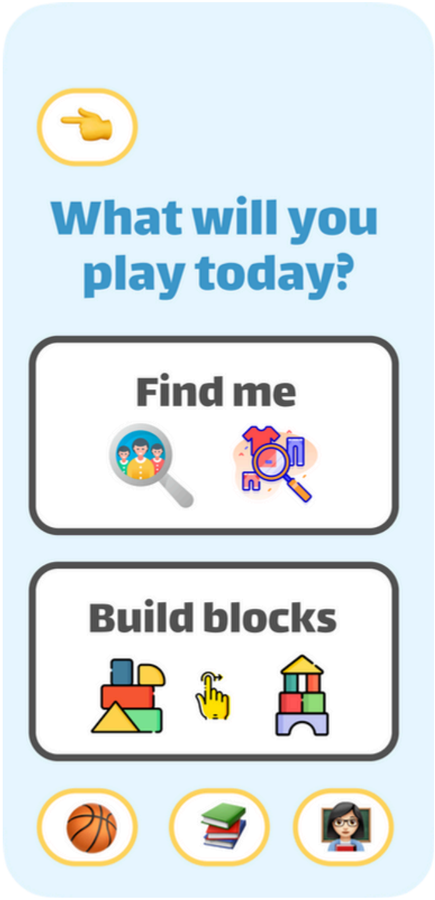

```python
# Example Usage
image1_path = "/content/drive/MyDrive/Colab Notebooks/RP/003/similarity/house 1.png"
image2_path = "/content/drive/MyDrive/Colab Notebooks/RP/003/similarity/house 2.png"

similarity, img1, img2 = compare_images(image1_path, image2_path)
print(f"Image Similarity: {similarity:.2f}%")
```

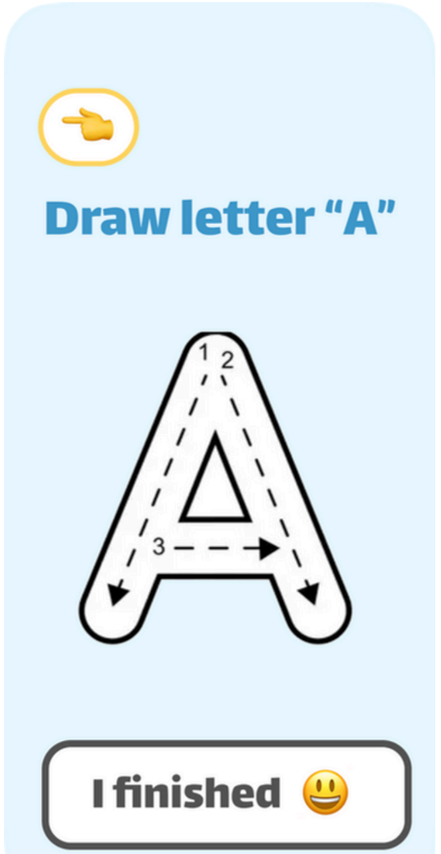
```
Image Similarity: 61.12%
```

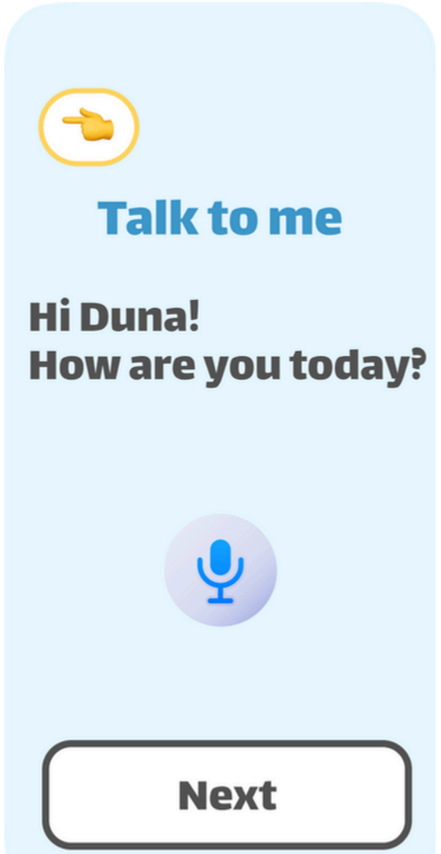
PROTOTYPE

Find the similarity between two images









OBJECTIVE COMPLETION STATUS

Identify the child's current cognitive and verbal communication skills.	100%
Analyze written letters from children to identify the current context.	100%
Predict the accuracy of hand-drawn letters for verbal assessment.	50%
Developing a game to building shapes from blocks to enhance cognitive skills.	25%
Employ DL models and rule-based systems for word prediction, accuracy of the letter and provide feedback.	25%

METHODOLOGY

Technologies

Flutter

Python

Firebase Database

TensorFlow

VS code

Figma

Scikit-learn

Techniques

Voice to text

Data Preprocessing

Pattern Recognition for Handwriting Analysis

Grammar Correction and Suggestions

Feature extraction and segmentation

Algorithms

SpaCy library (NLP)

CNN

- Conv2D
- MaxPooling2D

RNN (GRU)

Bidirectional LSTM

REQUIREMENT ANALYSIS - 1

Functional requirements

- Assess written letters accurately using trained models.
- DL model and rule-based system for word prediction and grammar correction.
- Choose predefined or custom exercises for writing and sentence structure.
- Predict new words according to the identified difficult sections.
- Track hours and identify difficulty areas.

Non-Functional requirements

- User-friendly Interfaces
- Quick processing speech recognition system.
- Realiability by accurate speech capturing
- Secure, ensuring the privacy of user data.
- Availability with high functionality and minimum downtime.

REQUIREMENT ANALYSIS - 2

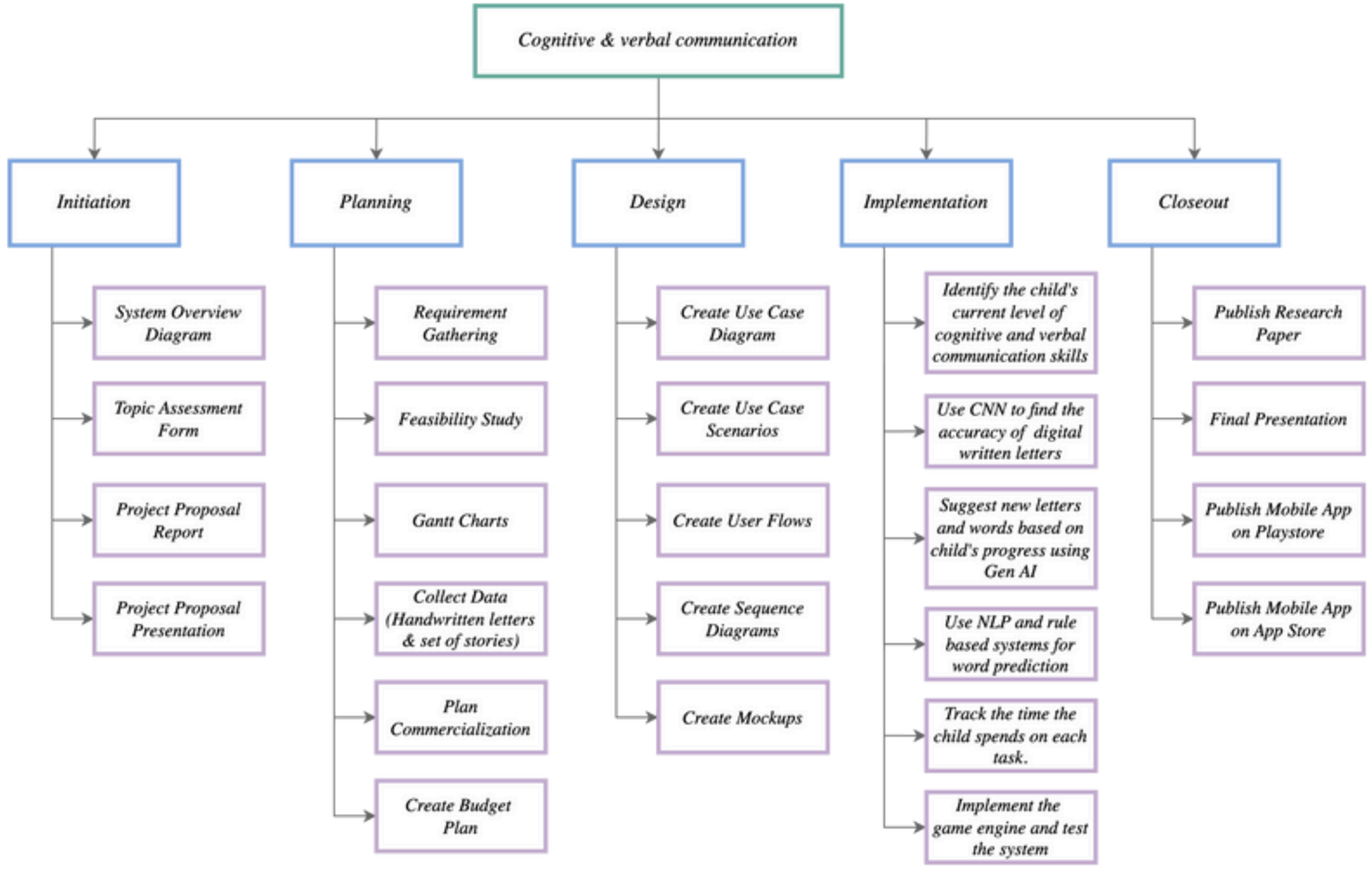
System requirements

- Compatibility varying screen sizes and performance capabilities.
- Adequate storage and memory to support the application
- The system should be compatible with major operating systems
- Dependencies - relevant ML libraries and game development tools.
- The system should be lightweight to run smoothly.

Personnel requirements

- Children with ASD
- Parents of children with ASD
- Dr. Asiri Hewamalage (@ FHB)
- Therapists associated with the therapy center we chose

WORK BREAKDOWN STRUCTURE



REFERENCES

[1] L. F. Guerrero-Vásquez et al., "Assessing Children's Perceptions of Live Interactions With Avatars: Preparations for Use in ASD Therapy in a Multi-Ethnic Context," in IEEE Access, vol. 8, pp. 168456-168469, 2020, doi: 10.1109/ACCESS.2020.3023737.

[2] Marvin, Alison & Marvin, Daniel & Lipkin, Paul & Law, Jessica. (2017). Analysis of Social Communication Questionnaire (SCQ) Screening for Children Less Than Age 4. Current Developmental Disorders Reports. 4. 10.1007/s40474-017-0122-1.

IMPROVING SPEECH AND LANGUAGE DEVELOPMENT

Umaira M M
IT21258312
[Software Engineering]

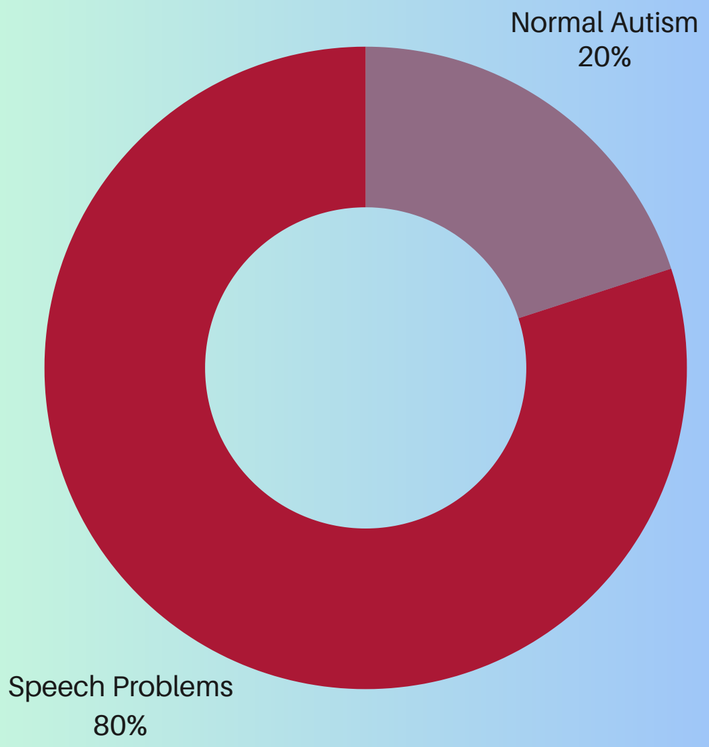
Background

Parents suffer from high levels of anxiety due to fear of child being isolated and unable to communicate.

Almost **all children with autism** retain **difficulty in speaking and language** [2].

some may have limited or no speech, while others might struggle with understanding and using language appropriately.

Main focus of the component is to make sure to improve the speech and language development of children in order make them adapt to normal living standards



In Sri Lanka **over 80 % of children with autism** presents with speech related problems [1].

Research Questions

How to improve speech and language skills of a child with autism ?

1 .How can we identify the current context of speech and language of a child with autism?

2. What Aspects are considered to identify and evaluate the current level and improvements?

3. What is the methodology use to enhance speech and language development?

4. How to identify if the pronunciation is bad using the proposed mechanism?

5. If an area of words are identified as weak pronunciation, how can it be improved?

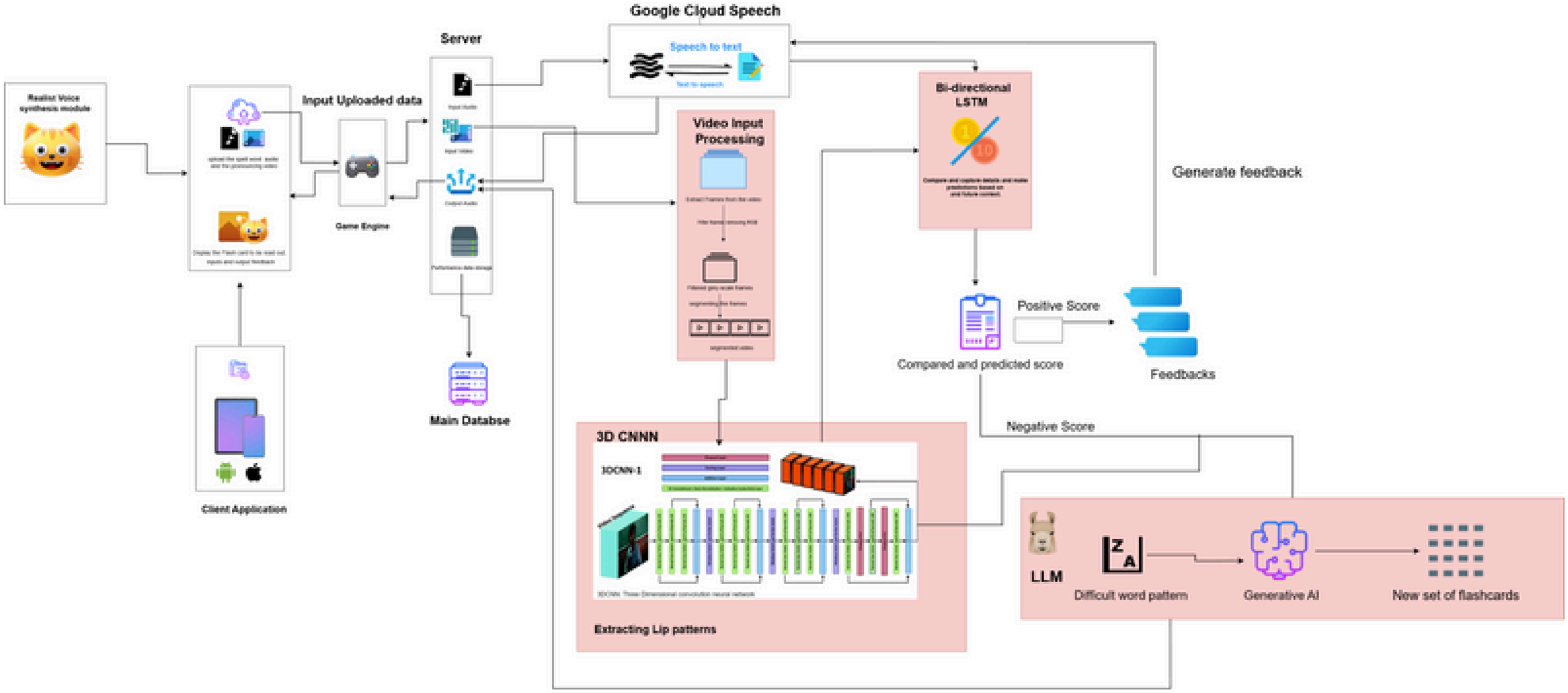
Main and Specific Objectives

Improvement of speech and language development

- Identifying and evaluating the child's current level of speech and language.
- Lip pattern Identification of the child when pronouncing a word.
- Developing an interactive flash card game with flash cards with words in different categories.
- Employing a pipeline to identify areas where the child is finding difficult to pronounce.
- Training the LLM to suggest new words using gen AI for the identified difficulty areas.

Methodology and Evidence of Completion

System Diagram



Objective 01 - Lip Pattern Identification when pronouncing a word or sentence

Evidence : Data set details and preprocessing techniques

Dataset consist of **1000 silent speaking** videos.

Data Divided into,

1. **Training**
2. **Testing**
3. **Validation**

```
train = data.take(450)
test = data.skip(450)
```

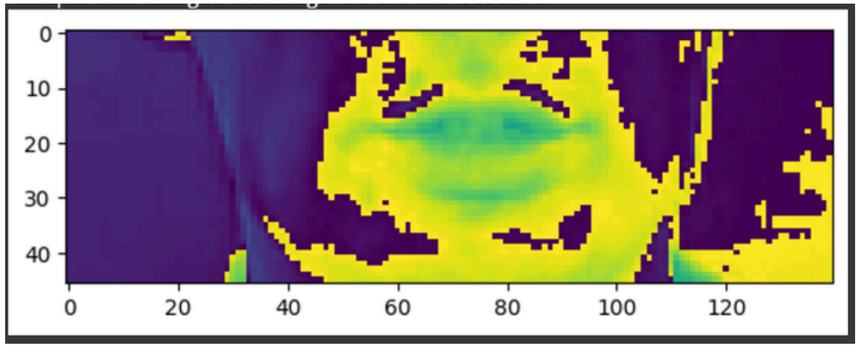
Pre Processing Techniques used,

1. **Time based frame extraction**
2. **Converting the frames to grey scale**
3. **Cropping the mouth region of each frame**
4. **Computing mean and standard deviation**
5. **Normalizing**

Objective 01 - Lip Pattern Identification when pronouncing a word or sentence

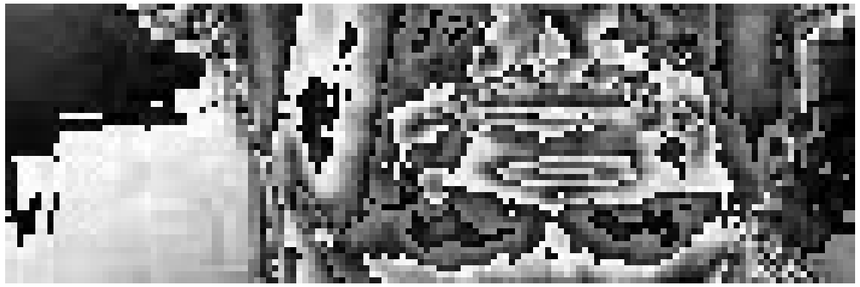
Evidence : Data preprocessing flow, processed output and model input

```python
def load_video(path:str) -> List[float]:
    cap = cv2.VideoCapture(path)
    frames = []
    for _ in range(int(cap.get(cv2.CAP_PROP_FRAME_COUNT))):
        ret, frame = cap.read()
        frame = tf.image.rgb_to_grayscale(frame)
        frames.append(frame[190:236,80:220,:])
    cap.release()

    mean = tf.math.reduce_mean(frames)
    std = tf.math.reduce_std(tf.cast(frames, tf.float32))
    return tf.cast((frames - mean), tf.float32) / std
```



Processed Frame

```python
import numpy as np
import imageio

frames = val[0][0]

frames_processed = [(frame.reshape(46, 140) * 255).astype(np.uint8) for frame in frames]

# Save as GIF
imageio.mimsave('./animation.gif', frames_processed, fps=10)
```



Processed GIF

Objective 01 - Lip Pattern Identification when pronouncing a word or sentence

Evidence : Codes of different models with different algorithms, libraries and frameworks

```
# Instantiate the model
model = Sequential()
# 1st ConvLSTM2D layer - combines Conv and LSTM in one
model.add(ConvLSTM2D(filters=128, kernel_size=(3, 3), input_shape=(75, 46, 140, 1),
                     padding='same', return_sequences=True))
model.add(Activation('relu'))
model.add(MaxPool3D(pool_size=(1, 2, 2)))
# 2nd ConvLSTM2D layer
model.add(ConvLSTM2D(filters=256, kernel_size=(3, 3), padding='same', return_sequences=True))
model.add(Activation('relu'))
model.add(MaxPool3D(pool_size=(1, 2, 2)))
# TimeDistributed Flatten to flatten across timesteps
model.add(TimeDistributed(Flatten()))
print("Shape after TimeDistributed Flatten: ", model.output_shape)
# TimeDistributed Dense layer to reduce dimensionality and reshape
model.add(TimeDistributed(Dense(units=128)))
print("Shape after TimeDistributed Dense: ", model.output_shape)
model.add(Reshape((75, 128)))
print("Shape after Reshape: ", model.output_shape)
# 1st Bidirectional LSTM layer
model.add(Bidirectional(LSTM(128, kernel_initializer=Orthogonal(), return_sequences=True)))
model.add(Dropout(0.5))
print("Shape after first Bidirectional LSTM: ", model.output_shape)
# 2nd Bidirectional LSTM layer
model.add(Bidirectional(LSTM(128, kernel_initializer=Orthogonal(), return_sequences=True)))
model.add(Dropout(0.5))
print("Shape after second Bidirectional LSTM: ", model.output_shape)
# Final Dense layer for vocabulary prediction
vocab_size = 40
model.add(Dense(vocab_size + 1, kernel_initializer=HeNormal(), activation='softmax'))
```

convLSTM model

```
model = Sequential() #instantialing the model
model.add(Conv3D(128, 3, input_shape=(75,46,140,1), padding='same')) #128 3DCNN model
model.add(Activation('relu')) # to give non-linearity to the neural network
model.add(MaxPool3D((1,2,2))) #takes the max values and condense them

model.add(Conv3D(256, 3, padding='same')) #repeat 3 times i.e, 3 layers
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))

model.add(Conv3D(75, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))

model.add(TimeDistributed(Flatten()))
print("Shape after TimeDistributed Flatten: ", model.output_shape)
# Adding after the TimeDistributed(Flatten()) layer
model.add(TimeDistributed(Dense(units=128)))  # Reduce dimensionality

# Adjust the Reshape layer to align with the label length (40 here):
model.add(Reshape(target_shape=(-1, 128)))  # You can adjust this shape based on model output

# Replacing LSTM with GRU
model.add(Bidirectional(GRU(128, kernel_initializer='Orthogonal', return_sequences=True)))
model.add(Dropout(.5))
print("Shape after first Bidirectional GRU: ", model.output_shape)
#Second GRU layer
model.add(Bidirectional(GRU(128, kernel_initializer='Orthogonal', return_sequences=True)))
model.add(Dropout(.5))
model.add(Dense(char_to_num.vocabulary_size()+1, kernel_initializer='he_normal', activation='softmax')) #softmax activati
print("Shape after Dense (final) layer: ", model.output_shape)
```

3DCNN + bidirectional GRU model

```
class My3DModel(nn.Module):
    def __init__(self):
        super(My3DModel, self).__init__()

        # 3D Convolutional Layers
        self.conv1 = nn.Conv3d(in_channels=1, out_channels=128, kernel_size=3, padding='same')
        self.pool1 = nn.MaxPool3d(kernel_size=(1, 2, 2))

        self.conv2 = nn.Conv3d(in_channels=128, out_channels=256, kernel_size=3, padding='same')
        self.pool2 = nn.MaxPool3d(kernel_size=(1, 2, 2))

        self.conv3 = nn.Conv3d(in_channels=256, out_channels=75, kernel_size=3, padding='same')
        self.pool3 = nn.MaxPool3d(kernel_size=(1, 2, 2))

        # Fully connected layer after flattening the output
        self.flatten = nn.Flatten()
        self.fc1 = nn.Linear(75 * 75 * 35, 128)  # Adjust the input size based on the output from the conv layers

        # Bidirectional LSTM Layers
        self.lstm1 = nn.LSTM(input_size=128, hidden_size=128, bidirectional=True, batch_first=True)
        self.dropout1 = nn.Dropout(0.5)

        self.lstm2 = nn.LSTM(input_size=256, hidden_size=128, bidirectional=True, batch_first=True)
        self.dropout2 = nn.Dropout(0.5)

        # Output Layer
        self.fc2 = nn.Linear(256, char_to_num.vocabulary_size() + 1)  # Adjusted to match the number of classes

    def forward(self, x):
        # Convolutional Layers
        x = self.pool1(F.relu(self.conv1(x)))
        x = self.pool2(F.relu(self.conv2(x)))
        x = self.pool3(F.relu(self.conv3(x)))

        # Flattening and passing through the fully connected layer
        x = self.flatten(x)
        x = self.fc1(x)

        # Reshape for LSTM layers
        x = x.view(x.size(0), -1, 128)  # Assuming input to LSTM is (batch_size, seq_len, input_size)

        # LSTM Layers
        x, _ = self.lstm1(x)
        x = self.dropout1(x)

        x, _ = self.lstm2(x)
        x = self.dropout2(x)

        # Output Layer
        x = self.fc2(x)

        return x

# Instantiate the model
model = My3DModel()
print(model)
```

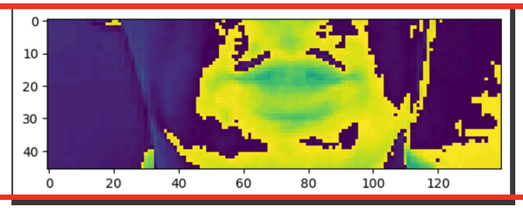
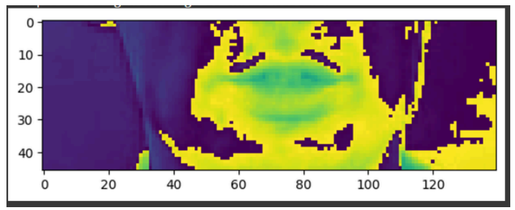
Pytorch + Scikit learn model
[3DCNN + bidirectional LSTM]

```
model = Sequential() #instantialing the model
model.add(Conv3D(128, 3, input_shape=(75,46,140,1), padding='same')) #128 3DCNN model
model.add(Activation('relu')) # to give non-linearity to the neural network
model.add(MaxPool3D((1,2,2))) #takes the max values and condense them

model.add(Conv3D(256, 3, padding='same')) #repeat 3 times i.e, 3 layers
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))

model.add(Conv3D(75, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))

model.add(TimeDistributed(Flatten()))
print("Shape after TimeDistributed Flatten: ", model.output_shape)
# Adding after the TimeDistributed(Flatten()) layer
model.add(TimeDistributed(Dense(units=128)))  # Reduce dimensionality

# Adjusting the Reshape layer to align with the label length (40 here):
model.add(Reshape(target_shape=(-1, 128)))

#2LSTM layers
model.add(Bidirectional(LSTM(128, kernel_initializer='Orthogonal', return_sequences=True)))
model.add(Dropout(.5))
print("Shape after first Bidirectional LSTM: ", model.output_shape)
#specifying bidirectional LSTM
model.add(Bidirectional(LSTM(128, kernel_initializer='Orthogonal', return_sequences=True)))
model.add(Dropout(.5)) #regularization, i.e., dropping out 50% of the units
model.add(Dense(char_to_num.vocabulary_size()+1, kernel_initializer='he_normal', activation='softmax'))
print("Shape after Dense (final) layer: ", model.output_shape)
```

3DCNN + bidirectional LSTM model

Objective 01 - Evidence

Selecting the Most Optimal Technology and Algorithms for training based on outputs

Algorithm	Framework /Library	Processed Frame	Training time	Loss Pattern	Prediction Accuracy Pattern
3DCNN + bidirectional LSTM	Tensorflow		8 h for 50 epochs	Steady Loss	Increased Gradually
3DCNN + bidirectional LSTM	Pytorch + scikit Image		-	-	-
3DCNN + bidirectional GRU	Tensorflow		7 h for 50 epochs	Inconsistence Loss	Inconsistent predictions

Objective 01 - Evidence

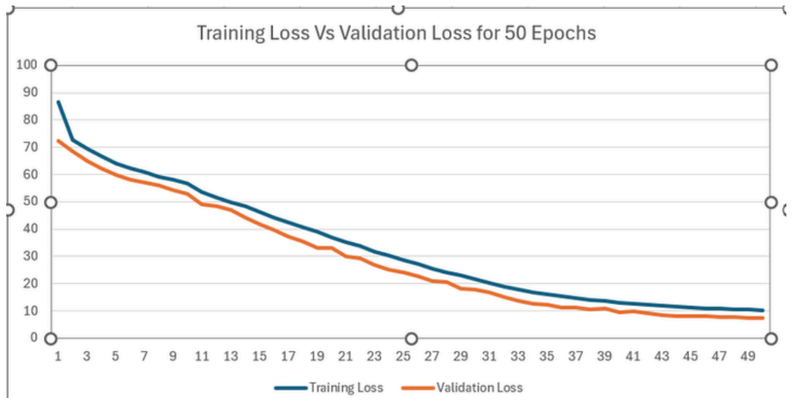
Fine tuning the model to select the most suitable state for predictions : Word Prediction Accuracy

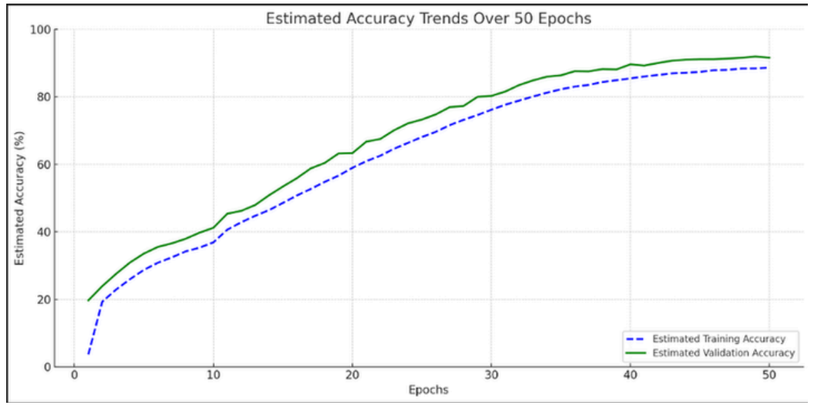
Epochs	No of Videos	Original no of words	Predicted words	Accuracy per video	Epoch Output Evidence
20	2	6 + 6 = 12	0 + 1 = 1	8.33%	450/450 [==========] - 616s 1s/step - loss: 41.3927 - val_loss: 35.9510 - lr: 1.0000e-04 Epoch 20/80 1/1 [==========] - 0s 189ms/step Original: set green by v eight now Prediction: plac ble it o o Original: lay blue at k one soon Prediction: plac blue by io ow
40	2	6 + 6 = 12	4 + 3 = 7	58.3%	450/450 [==========] - 626s 1s/step - loss: 15.5941 - val_loss: 11.7606 - lr: 4.0657e-05 Epoch 40/80 1/1 [==========] - 0s 190ms/step Original: lay red by l zero now Prediction: lay red by er now Original: lay green in f three again Prediction: lay gren in tre again
50	2	6 + 6 = 12	4 +5 = 9	75%	450/450 [==========] - 630s 1s/step - loss: 12.0018 - val_loss: 8.9994 - lr: 1.6530e-05 Epoch 49/80 1/1 [==========] - 0s 185ms/step Original: set red at u four please Prediction: set red at fou please Original: bin blue at z six please Prediction: bin blue at six please
80	2	6 + 6 = 12	5 + 5 = 10	83.33%	450/450 [==========] - 624s 1s/step - loss: 9.9459 - val_loss: 6.9327 - lr: 7.4466e-07 Epoch 80/80 1/1 [==========] - 0s 193ms/step Original: place green at y one soon Prediction: place gren at one son Original: lay red with f two please Prediction: lay red with two please

Objective 01 - Evidence

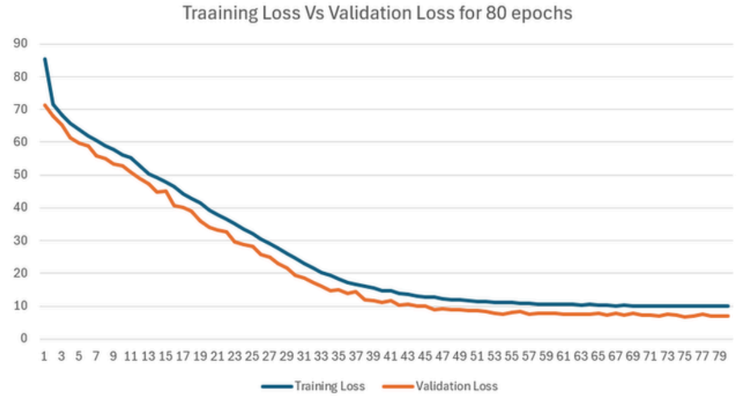
Fine tuning the model to select the most suitable state for predictions : Loss and Accuracy Graphs

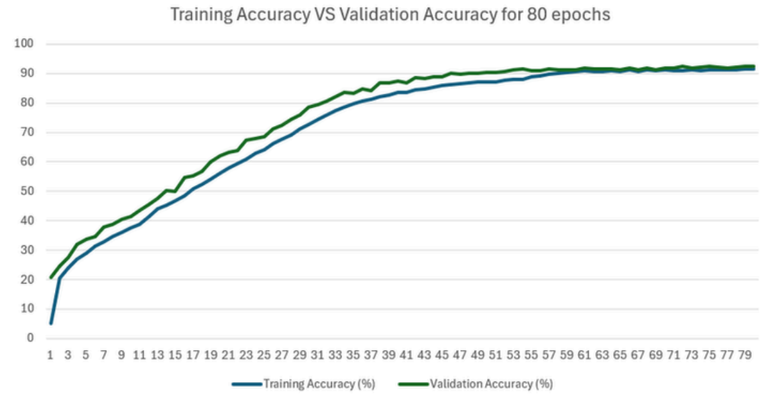
50 Epochs





80 Epochs

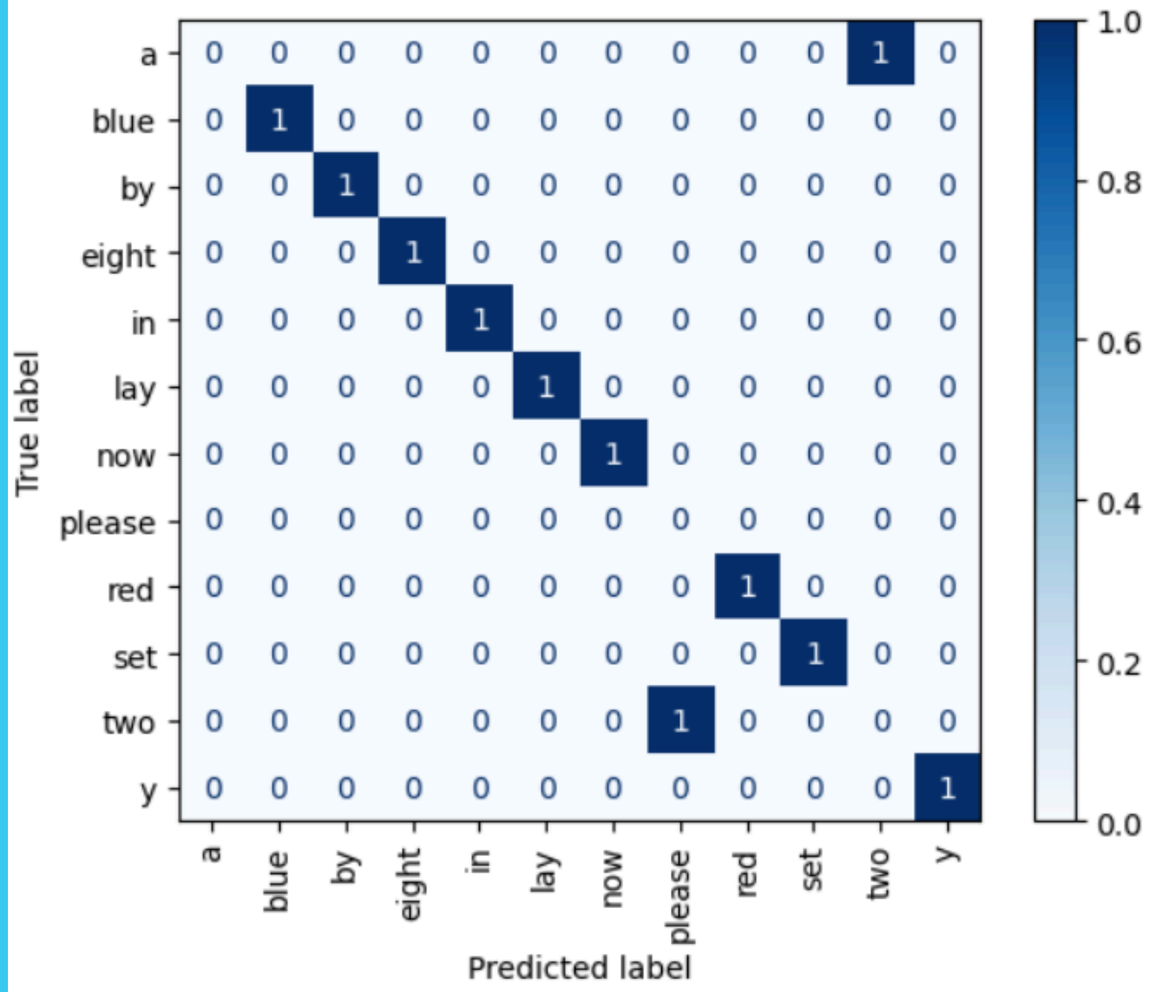




Signs of Overfitting

Objective 01 - Evidence

Confusion Metrics for 50 epochs



Test cases for predictions at 50 epochs

Test Case Id	01
Test Case	Lip pattern Analysis
Test Scenario	Predicting the sentence in the test video at epoch 50
Precondition	Model should be trained, and videos should be pre-processed
Input	Video of a man reading out a sentence
Expected Output	Sentence: bin blue at g eight now
Actual Result	bin blue at g eight now
Status (Pass/ Fail)	Pass

Test Case Id	02
Test Case	Lip pattern Analysis
Test Scenario	Predicting the sentence in the test video at epoch 50
Precondition	Model should be trained, and videos should be pre-processed
Input	Video of a man reading out a sentence
Expected Output	Sentence: place green at y one soon
Actual Result	Place gren at one soo n
Status (Pass/ Fail)	Fail

Objective 02 - Generating similar words and images using genAI or an LLM

2.1 Generating similar texts - Evidence of code bases and updates

app.py

pythonBackend > app.py > ...

```python
from flask import Flask, request, jsonify
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch

app = Flask(__name__)

# Load GPT-2 model and tokenizer
model_name = "gpt2"
tokenizer = GPT2Tokenizer.from_pretrained(model_name)
model = GPT2LMHeadModel.from_pretrained(model_name)

def generate_similar_words(word):
    input_text = f"Suggest words that sound similar to: {word}. "
    input_ids = tokenizer.encode(input_text, return_tensors='pt')

    # Generate output from the model
    with torch.no_grad():
        output = model.generate(input_ids, max_length=50, num_return_sequences=1, temperature=0.7)

    generated_text = tokenizer.decode(output[0], skip_special_tokens=True)
    similar_words = generated_text.replace(input_text, '').strip().split(',')

    return [w.strip() for w in similar_words]

@app.route('/generate', methods=['POST'])
def generate():
    data = request.json
    word = data.get("word", "")
    if not word:
        return jsonify({"error": "No word provided"}), 400
    similar_words = generate_similar_words(word)
    return jsonify({"similar_words": similar_words})

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

Initial function

```python
from flask import Flask, request, jsonify
import pronouncing

app = Flask(__name__)

@app.route('/generate', methods=['POST'])
def generate_similar_words():
    data = request.json
    word = data.get('word')

    if not word:
        return jsonify({"error": "No word provided"}), 400

    # Get rhyming words using the pronouncing library
    rhymes = pronouncing.rhymes(word)

    # If no rhymes are found, return a default message
    if not rhymes:
        rhymes = ["No rhyming words found"]

    # Limit to 5 rhyming words
    rhymes = rhymes[:5]

    response = {
        "original_word": word,
        "rhyming_words": rhymes
    }

    return jsonify(response)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

Updated function with the library

app.py

pythonBackend > app.py > ...

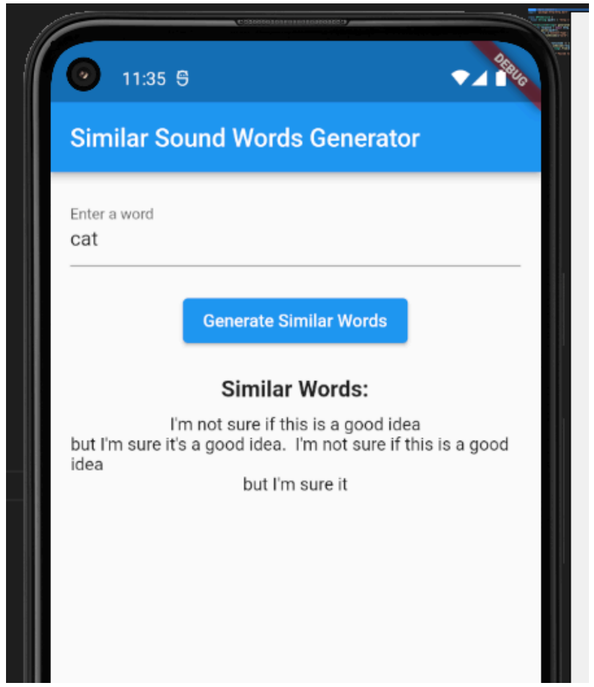
```python
from flask import Flask, request, jsonify
import pronouncing

app = Flask(__name__)

# List of common, simple words suitable for preschool children
simple_words = set([
    "cat", "bat", "hat", "dog", "log", "sun", "run", "fun", "car", "star", "cow", "bow",
    "ball", "fall", "tree", "bee", "bird", "fish", "frog", "apple", "banana", "grape",
    "juice", "milk", "toy", "book", "shoe", "sock", "pen", "desk", "mat", "sand", "moon","call", "silk", "drink", "blink", "see"
])

@app.route('/generate', methods=['POST'])
def generate_similar_words():
    data = request.json
    word = data.get('word')

    if not word:
        return jsonify({"error": "No word provided"}), 400

    # Get rhyming words using the pronouncing library
    rhymes = pronouncing.rhymes(word)

    # If no rhymes are found, return a default message
    if not rhymes:
        rhymes = ["No rhyming words found"]

    # Limit to 5 rhyming words and filter to simple words
    filtered_rhymes = [rhyme for rhyme in rhymes if rhyme in simple_words][:5]

    if not filtered_rhymes:
        filtered_rhymes = ["No rhyming words found"]

    response = {
        "original_word": word,
        "rhyming_words": filtered_rhymes
    }

    return jsonify(response)

if __name__ == '__main__':
    # Run the app on all available IP addresses (0.0.0.0) and port 5000
    app.run(host='0.0.0.0', port=5000, debug=True)
```

Final function with filters

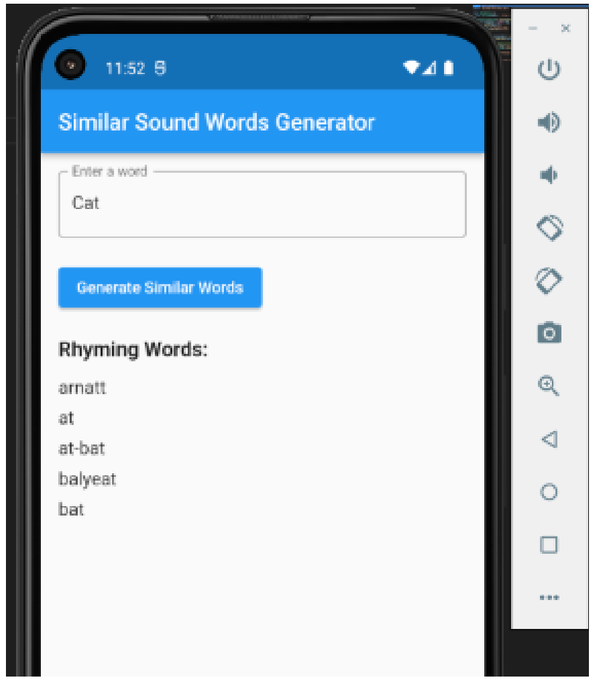
GPT-2 model hosted locally via a python backend

Objective 02 - Evidence

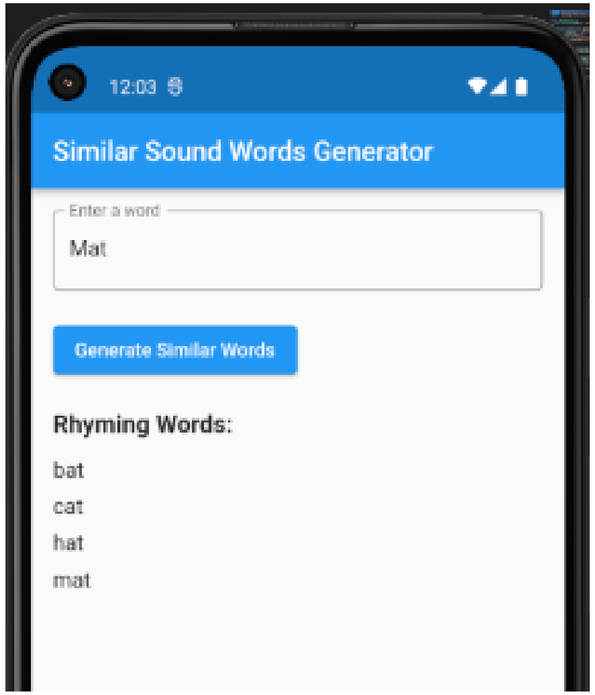
2.1 Generating similar texts : Outputs



Initial Output



Output after adding the **pronunciation library** for rhyming words



Output after **adding the filter** to filter words for kids

Objective 02 - Evidence

2.2 Generating images for a text : Evidence of code and updates

lib > services > image_service.dart > generateImageForWord

```dart
import 'dart:convert';
import 'package:http/http.dart' as http;

Future<String> generateImageForWord(String word) async {
  final apiKey =
      'r8_4qW3goaNV5ikRvIXOA82BR87cjrKRwm1ryNCt'; // Replica
  final response = await http.post(
    Uri.parse('https://api.replicate.com/v1/predictions'),
    headers: {
      'Authorization': 'Token $apiKey',
      'Content-Type': 'application/json',
    },
    body: jsonEncode({
      'version': 'latest',
      'input': {'prompt': word},
    }),
  );

  return jsonDecode(response.body)['output'][0];
}
```

```dart
import 'dart:typed_data';
import 'package:http/http.dart' as http;

class ImageService {
  final String _apiToken = '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';
  final String _modelEndpoint =
      'https://api-inference.huggingface.co/models/black-forest-labs/FLUX.1-schnell';

  Future<Uint8List> generateImage(String word) async {
    final Uri url = Uri.parse(uri: _modelEndpoint);

    final Map<String, String> headers = <String, String>{
      'Authorization': 'Bearer $_apiToken',
      'Content-Type': 'application/json',
    };

    final String body = '{"inputs": "$word"}'; // Pass word as input text

    try {
      final Response response = await http.post(
        url: url,
        headers: headers,
        body: body,
      );

      if (response.statusCode == 200) {
        // If the response is an image, return the raw bytes
        return response.bodyBytes;
      } else {
        throw Exception(message: 'Failed to generate image: ${response.statusCode}');
      }
    } catch (e) {
      throw Exception(message: 'Error generating image: $e');
    }
```

Attempt 01

Gen AI Model : Stable Diffusion by Stability AI [3]

API via : Replicate.com [5] and API token

Attempt 02

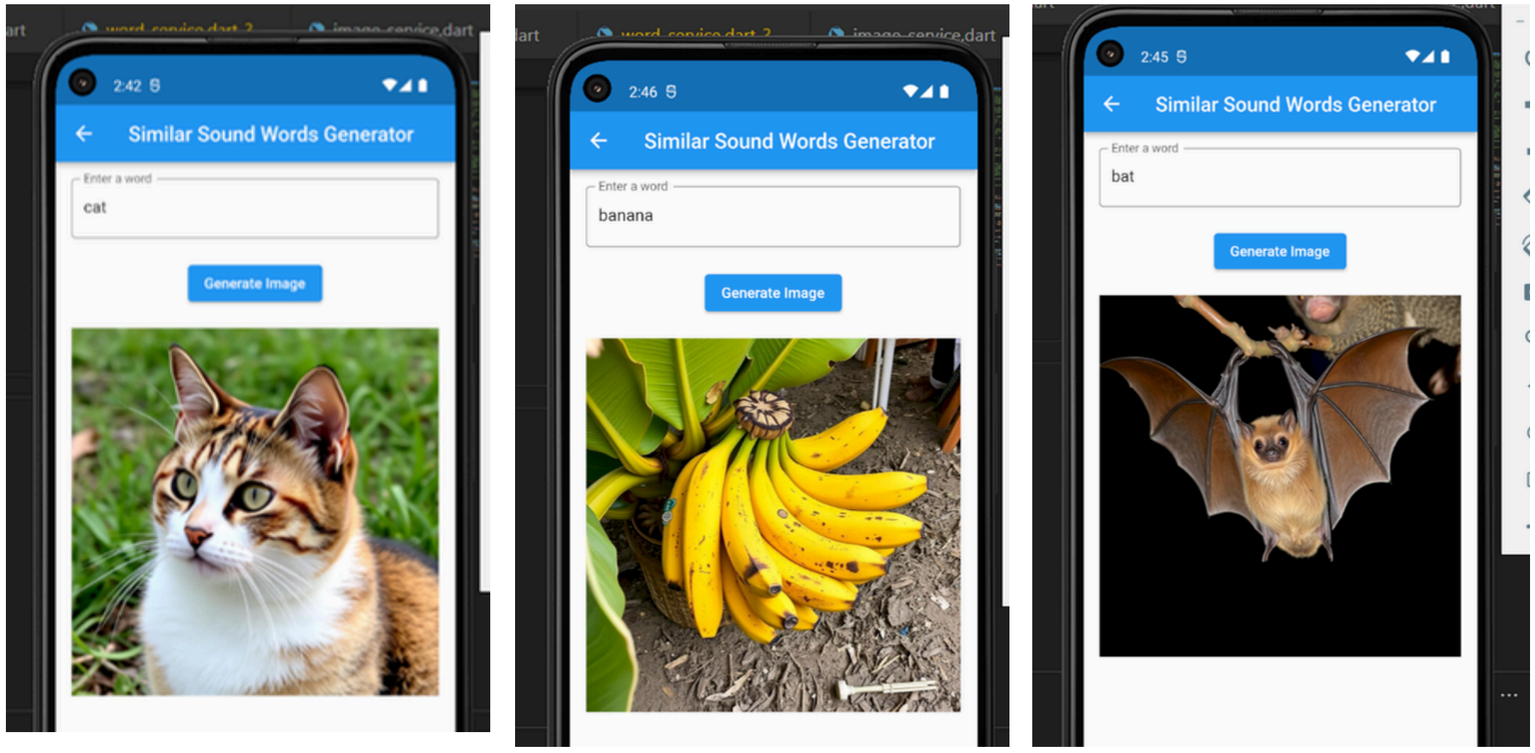
Gen AI Model : Flux model by Blackforest AI [4]

API via : Huggingface.com [6] and API token

Objective 02 - Evidence

2.2 Generating images for texts : Outputs

Outputs obtained from the Flux model by Blackforest AI [4].



Why real-life images ?

"Children with ASD achieved significantly greater retention accuracy when learning from photographs rather than cartoons [7] ."

Objective 03 and 04 - Partial Completion Evidence

Code base of creating a pipeline to predict for an external source

```python
import tensorflow as tf
import numpy as np
import imageio
import os

def predict_from_external_video_refined(video_url, model, num_to_char):
    """
    Loads a video from a URL, preprocesses frames, and predicts the lip-reading output.

    Args:
        video_url: Path to the external video.
        model: The trained lip-reading model.
        num_to_char: Function to map numerical predictions to characters.

    Returns:
        Predicted lip-reading output as a string, or None if there was an error.
    """
    try:
        # Load video frames
        frames = load_video(video_url)  # Ensure this function returns a tensor of shape (75, 46, 140, 1)
        if frames is None:
            print("Failed to load or process the video.")
            return None

        # Ensure frames are in the correct format for prediction
        frames = tf.convert_to_tensor(frames, dtype=tf.float32)
        print("External video shape:", frames.shape)
        input_frames = tf.expand_dims(frames, axis=0)  # Add batch dimension

        # Perform prediction
        yhat = model.predict(input_frames)

        # Determine the sequence length dynamically based on the model's output
        sequence_length = [yhat.shape[1]]

        # Decode the predictions using CTC
        decoded = tf.keras.backend.ctc_decode(yhat, input_length=sequence_length, greedy=True)[0][0].numpy()
        predicted_text = [
            tf.strings.reduce_join([num_to_char(word) for word in sentence]) for sentence in decoded
        ]

        # Convert to string output
        return predicted_text[0].numpy().decode("utf-8")
    except Exception as e:
        print(f"Error during prediction: {e}")
        return None

# Example usage
external_video_url = "/content/redBlueGreen.mp4"
prediction = predict_from_external_video_refined(external_video_url, model, num_to_char)
if prediction:
    print(f"Predicted Lip Reading: {prediction}")
```

```
External video shape: (35, 46, 140, 1)
1/1 [==============================] - 1s 815ms/step
Predicted Lip Reading: wet wr wt wo n
```

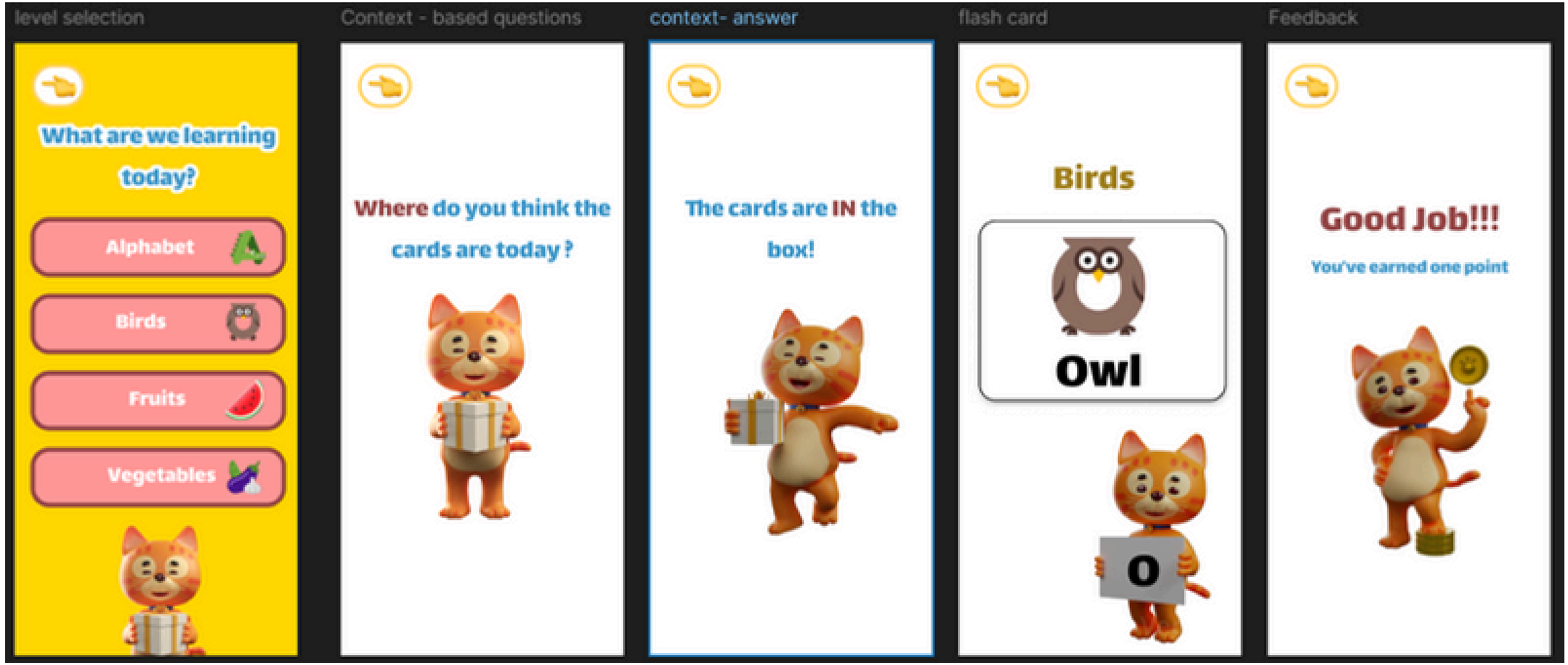
Initial pipeline with incorrect prediction

```python
import numpy as np

def adjust_video_frames(video_frames, required_frame_count=75):
    """
    Adjusts the frame count of the video by padding or duplicating frames.

    Args:
        video_frames: A numpy array of shape (current_frame_count, height, width, channels).
        required_frame_count: The desired number of frames (default: 75).

    Returns:
        A numpy array of shape (required_frame_count, height, width, channels).
    """
    current_frame_count = video_frames.shape[0]

    if current_frame_count < required_frame_count:
        # Pad frames: Repeat the last frame until the required frame count is reached
        padding_frames = required_frame_count - current_frame_count
        last_frame = video_frames[-1]  # Get the last frame
        pad = np.repeat(last_frame[np.newaxis, ...], padding_frames, axis=0)  # Repeat last frame
        adjusted_frames = np.concatenate([video_frames, pad], axis=0)
    elif current_frame_count > required_frame_count:
        # Crop frames: Take only the first required_frame_count frames
        adjusted_frames = video_frames[:required_frame_count]
    else:
        # No adjustment needed
        adjusted_frames = video_frames

    return adjusted_frames
```

Added Function to reshape the input frames of external videos if the frame shape is different

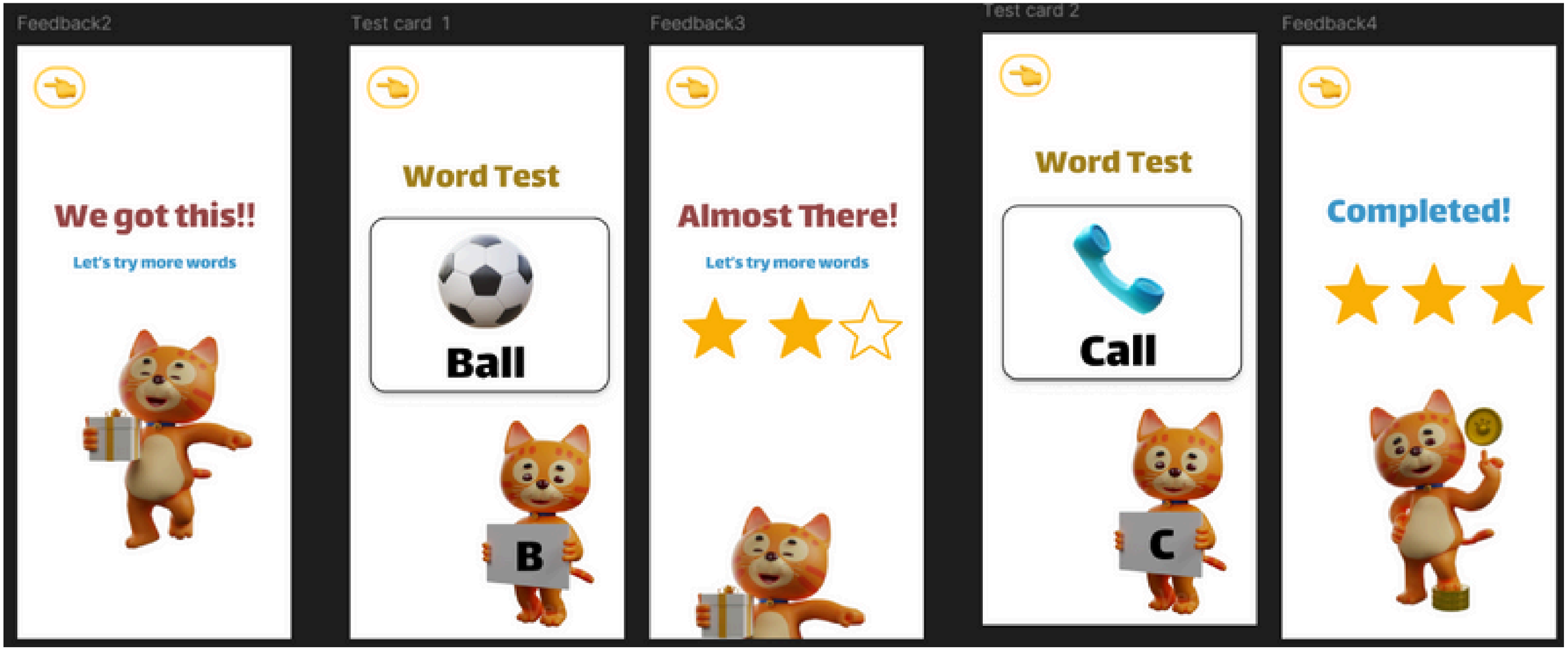
Designed Figma Prototypes - Flow 1

These prototypes were shown to the desired user groups during field visits



Designed Figma Prototypes - Flow 2

These prototypes were shown to the desired user groups during field visits



Objective Completion Status

	Status
Lip pattern Identification when pronouncing a word or sentence	Completed
Training the LLM to suggest new words using gen AI for the identified difficulty areas.	Completed
Employing a pipeline to identify areas where the child is finding difficult to pronounce.	40 %
Identifying and evaluating the child's current level of speech and language.	50 %
Developing an interactive flash card game with flash cards with words in different categories.	Haven't started yet

Tools Technologies and Algorithms

Technologies

Flutter
Python
Tensorflow
OpenCv
Scikit-learn
Google Colab
VS code
Hugging Face API

Techniques

Feature extraction and segmentation

Augmentation

Data Preprocessing

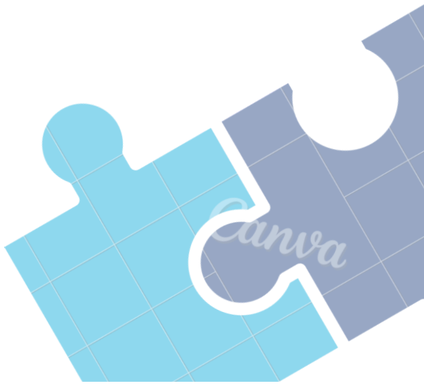
Time based feature extraction

Algorithms and Functions

3DCNN

Bi-directional LSTM

Softmax Activation Function



Requirement Analysis

Functional requirements

- should be able to recognize speech accurately
- should accurately capture lip movements while pronouncing.
- should capture the words the child is finding difficult to pronounce
- should predict new words according to the identified difficult sections.
- The games should be short and limited to certain time to avoid screen addiction.

Non-Functional requirements

- User-friendly Interfaces
- Quick processing speech recognition system.
- Realiability by accurate speech capturing
- Secure, ensuring the privacy of user data.
- Availability with high functionality and minimum downtime.

Requirement Analysis

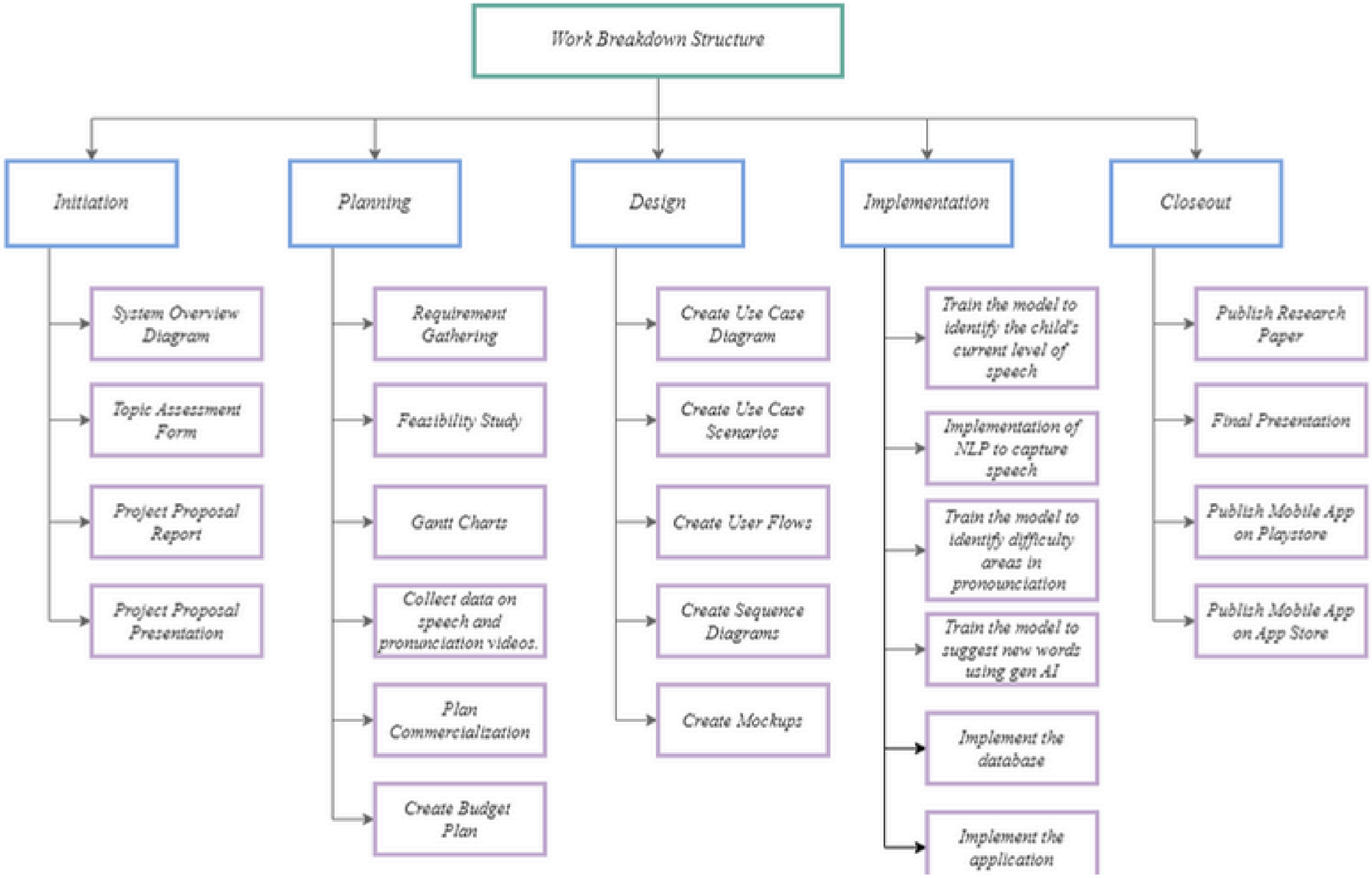
System requirements

- A device with a camera and microphone
- Adequate storage and memory to support the application
- The system should be compatible with major operating systems
- Dependencies - relevant ML libraries and game development tools.
- A strong network connection to set up and update.

Personnel requirements

- Children with autism who has speech difficulties.
- Parents and caregivers of children.
- Dr. Asiri Hewamalage

Work Breakdown Structure



References

[1] P. H. Perera, "Reasons for which parents seek help for their children with autism," in Management of Children with Special Needs : Manual for Primary Health Care Workers in Sri Lanka, Colombo, Sri Lanka., Ministry of Health Sri Lanka, p. 13.

[2] M. Santiputri, E. B. Sembiring, N. Z. Janah and M. K. Mufida, "Abangmanis: Speech Theraphy for Autism Monitoring Mobile Application," 2019 2nd International Conference on Applied Engineering (ICAE), Batam, Indonesia, 2019, pp. 1-6, doi: 10.1109/ICAE47758.2019.9221735. keywords: {Autism;speech therapy;monitoring;multimedia;mobile},

[3] "stability.ai," [Online]. Available: https://stability.ai/.

[4] "replicate.com," [Online]. Available: https://replicate.com/stability-ai/sdxl.

[5] "github.com," [Online]. Available: https://github.com/black-forest-labs/flux.

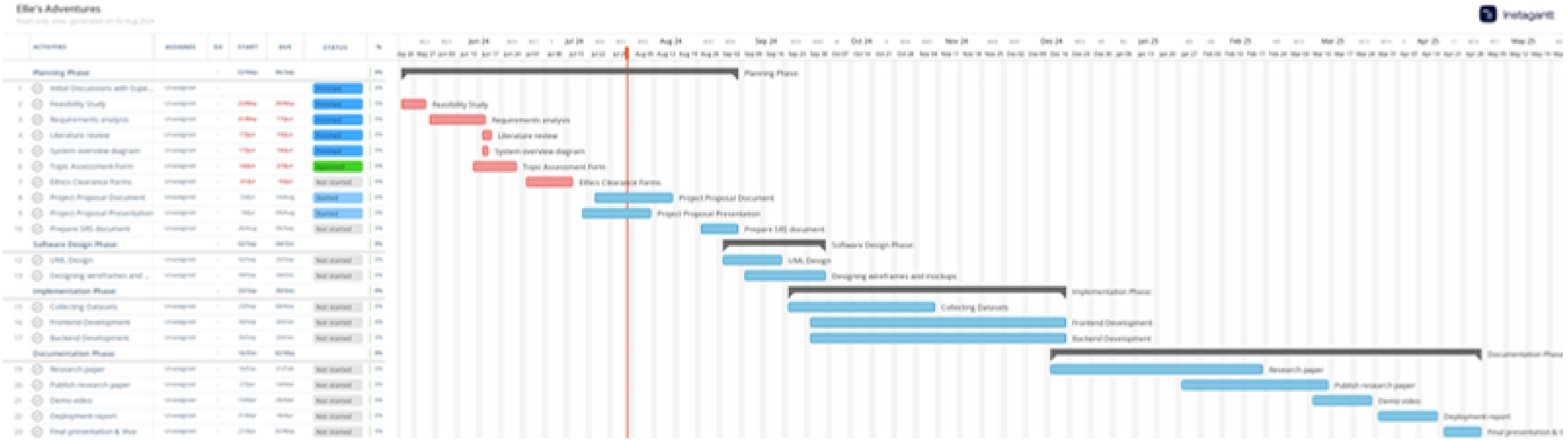
[6] "huggingface.co," [Online]. Available: https://huggingface.co/black-forest-labs/FLUX.1-schnell.

[7] C. H. Cheriece K Carter, "Are Children With Autism More Likely to Retain Object Names When Learning From Colour Photographs or Black-and-White Cartoons?," PubMedCentral, 2021.

Software Features

- Customization and Personalized UI/UX
- Layered client-server architecture
- Sensory-Friendly Design
- Deploy as a distributed system
- Multi-tenant encrypted storage
- Linting code using tools such as Dart Linting for Flutter and PyLint for Python
- Google confidential computing
- Unit testing for Flutter using flutter_test

Ghantt Chart

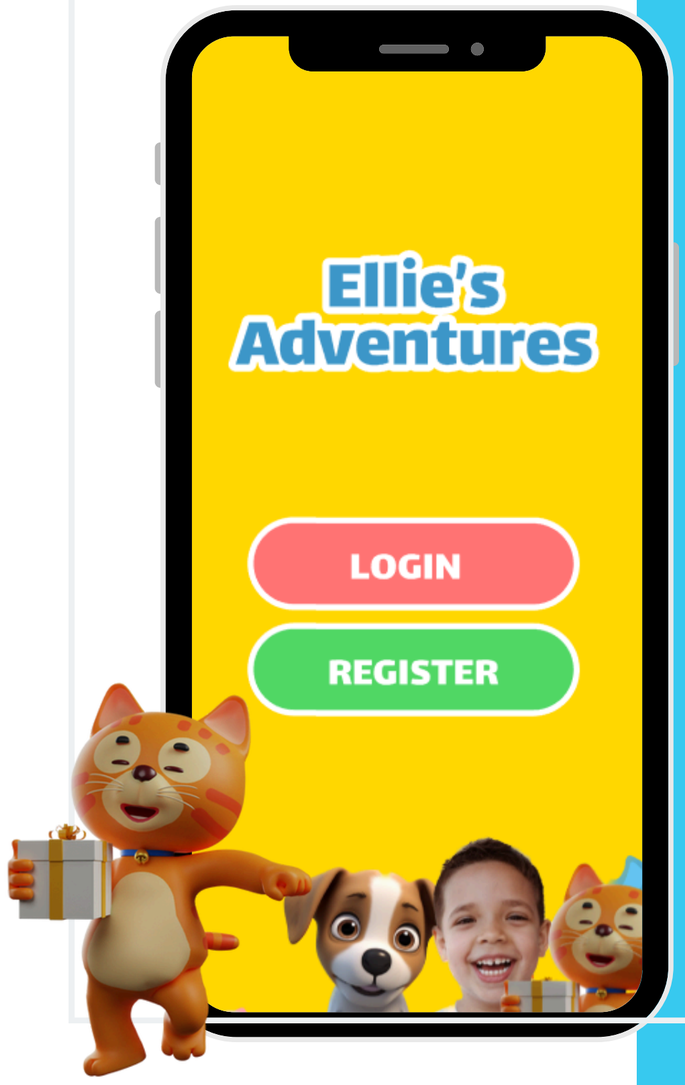


Budget

Initiative	Amount (lkr)
Travelling cost for data collection	30 000
Cost of deployment to cloud	25 000
Cost of storage and database	7 000
ML model training and deployment	35 000
Cost of hosting in play store	8 000
ccost of hosting in app store	33 000 / year

Commercialization

Free Package - LKR 0.00	Premium Package - LKR 4000
• Limited real-time feedback • Limited GenAI features • 5 free genAI credits daily • 5 free exercises daily • No real-time AI features • Limits on customizing the 3d avatar • Limits on voice selection for the avatar	• Unlimited access to the application • More personalized feedbacks



References

[1] https://www.who.int/news-room/fact-sheets/detail/autism-spectrum-disorders, Health Organization," [Online]. Available: "World https://www.who.int/news-room/fact sheets/detail/autism-spectrum-disorders.

[2] A. H. Vanniarachchy, "Autism in Sri Lanka," Ceylon Today, [Online]. Available: https://ceylontoday.lk/2024/07/20/autism-in-sri-lanka/.

[3] P. H. Perera, "Reasons for which parents seek help for their children with autism," in Management of Children with Special Needs : Manual for Primary Health Care Workers in Sri Lanka, Colombo, Sri Lanka., Ministry of Health Sri Lanka, p. 13.

[4] C. J. Zampella, L. A. L. Wang, M. Haley, A. G. Hutchinson and A. d.Marchena, “Motor Skill Differences in Autism Spectrum Disorder: a Clinically Focused Review,” Current psychiatry reports, 2021.

Thank You !