# Applying Object Detection to watch in-game ads

BY

Awwal Alagbe and Temidire Dimowo

# Outline

Introduction (and Motivation)

Problem of Study

Related work

Methodology

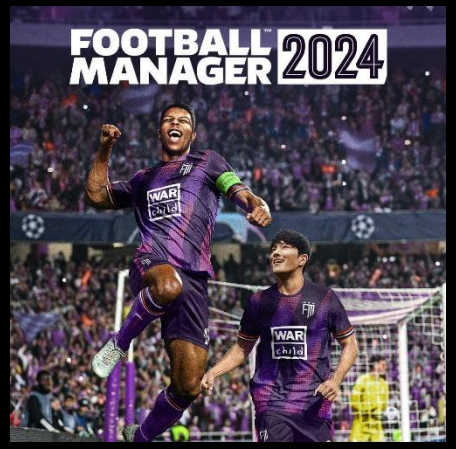
Experiments and Results

Discussion/Limitations

Conclusion/Future Work

Contribution of each of the team members

# Introduction (and Motivation)

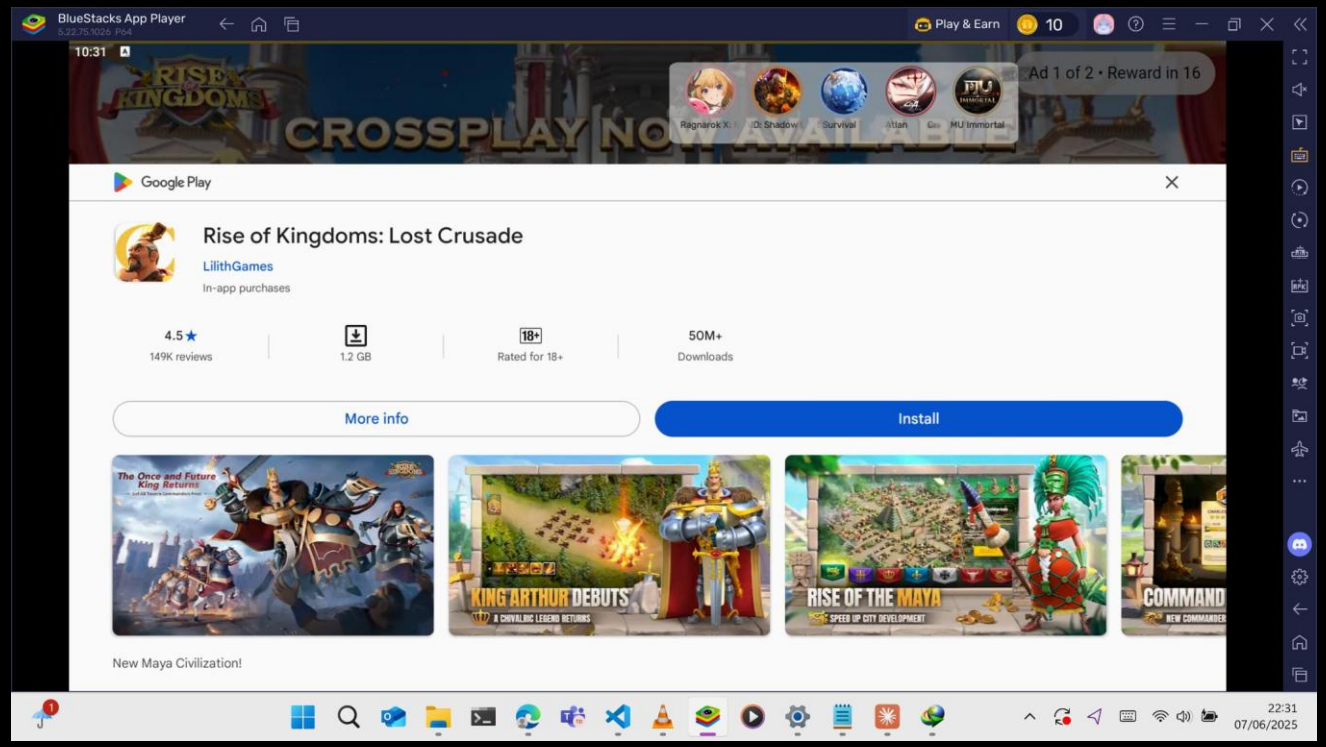




# Introduction (and Motivation)


**3 hours 30 minutes per day**

| | |
|---|---|
| Watching Ads to earn resources | Playing matches |
| | Training Players |

# Problem of the study




**Identifying and interacting with in-game elements**

## Close and Skip buttons

# Related Works

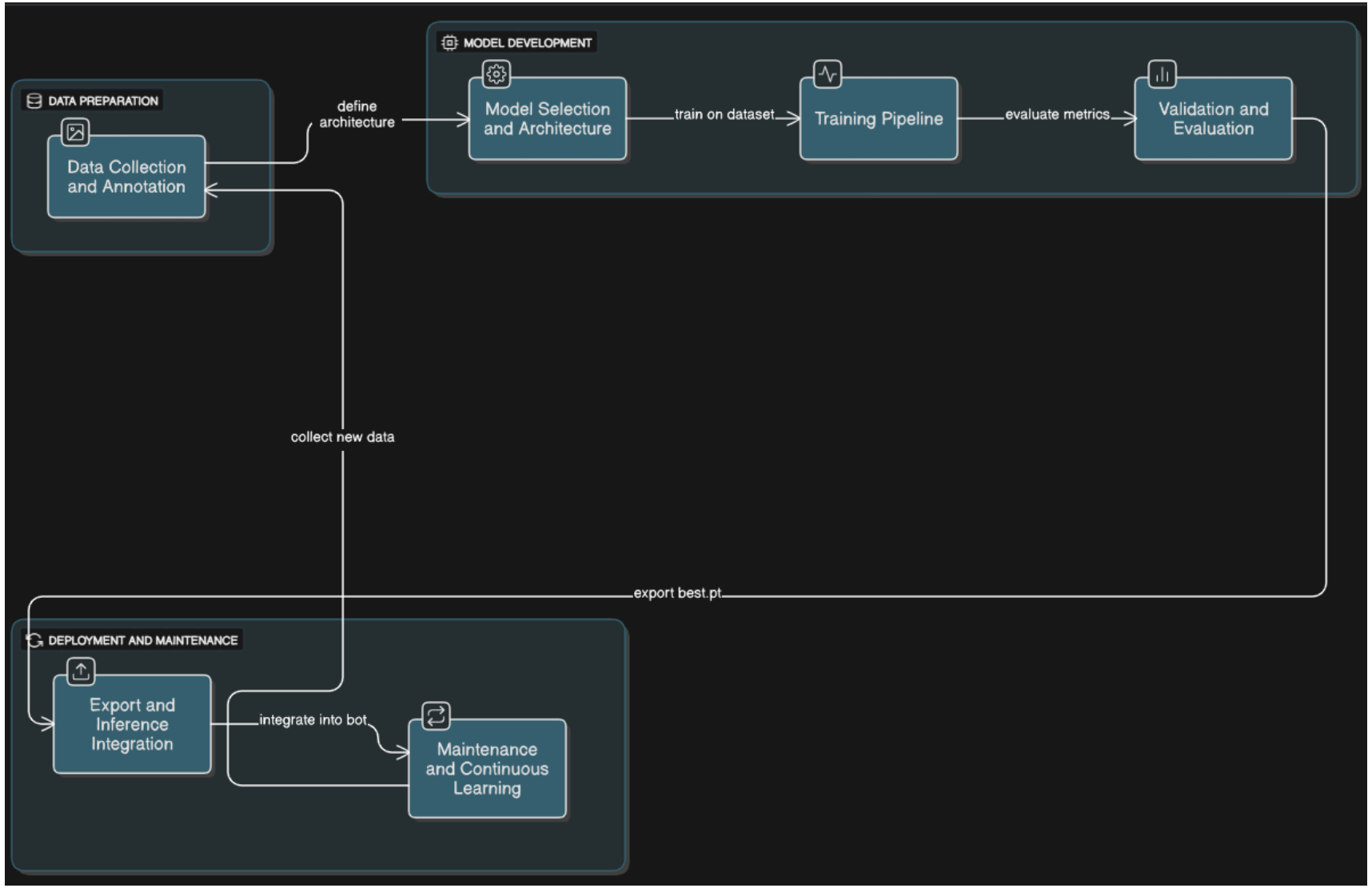
Existing Screen-Based Automation

1. Primarily uses template matching or hard-coded coordinates
2. Lacks Flexibility

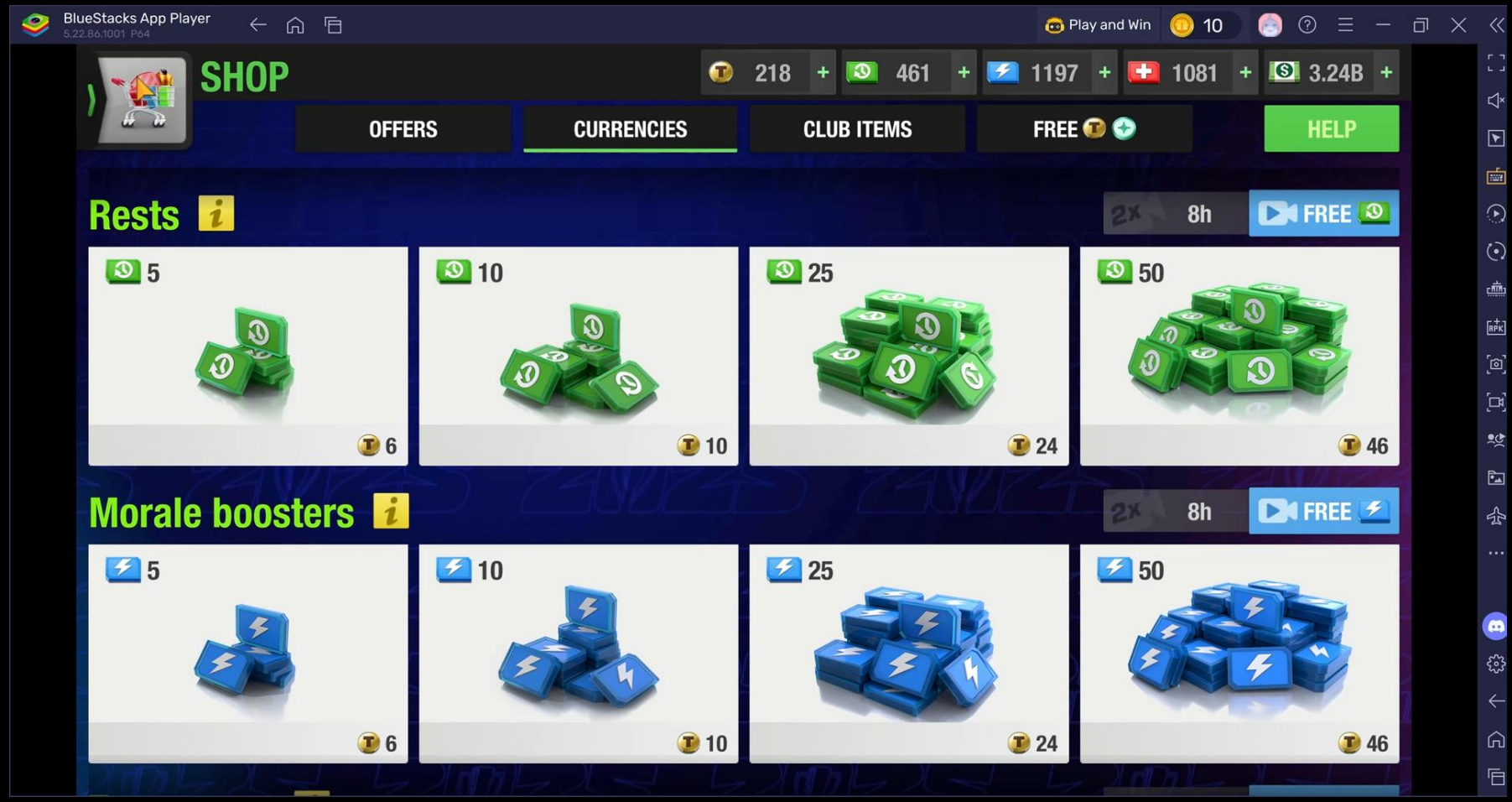
YOLO-based GUI automation

1. Some attempts at YOLO-based GUI automation exist but are limited in scope and emulator integration
2. Pure rule-based systems fail with UI changes
3. Full UI-testing frameworks are heavy and not optimized for real-time use

# Methodology



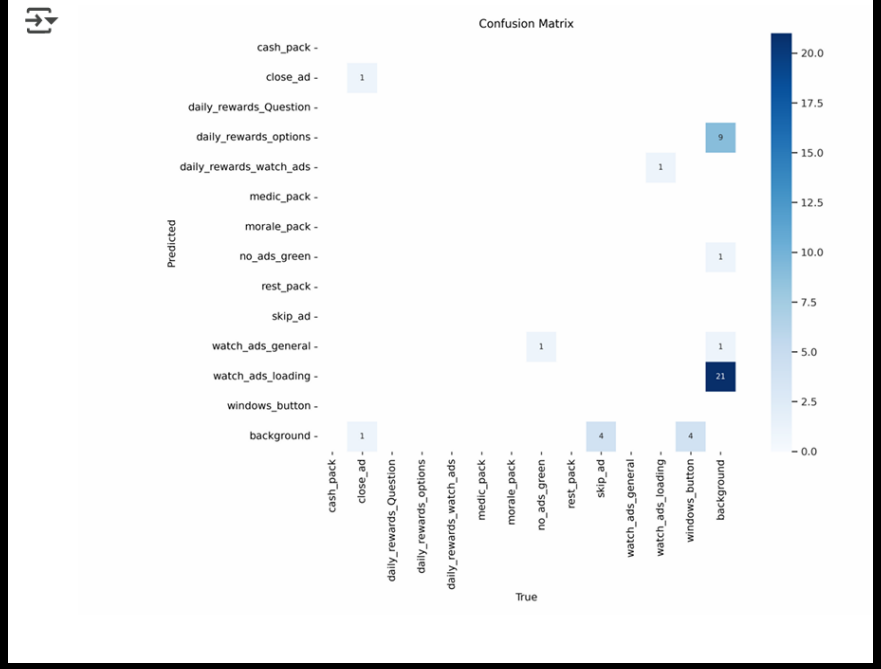
# Experiments and Results



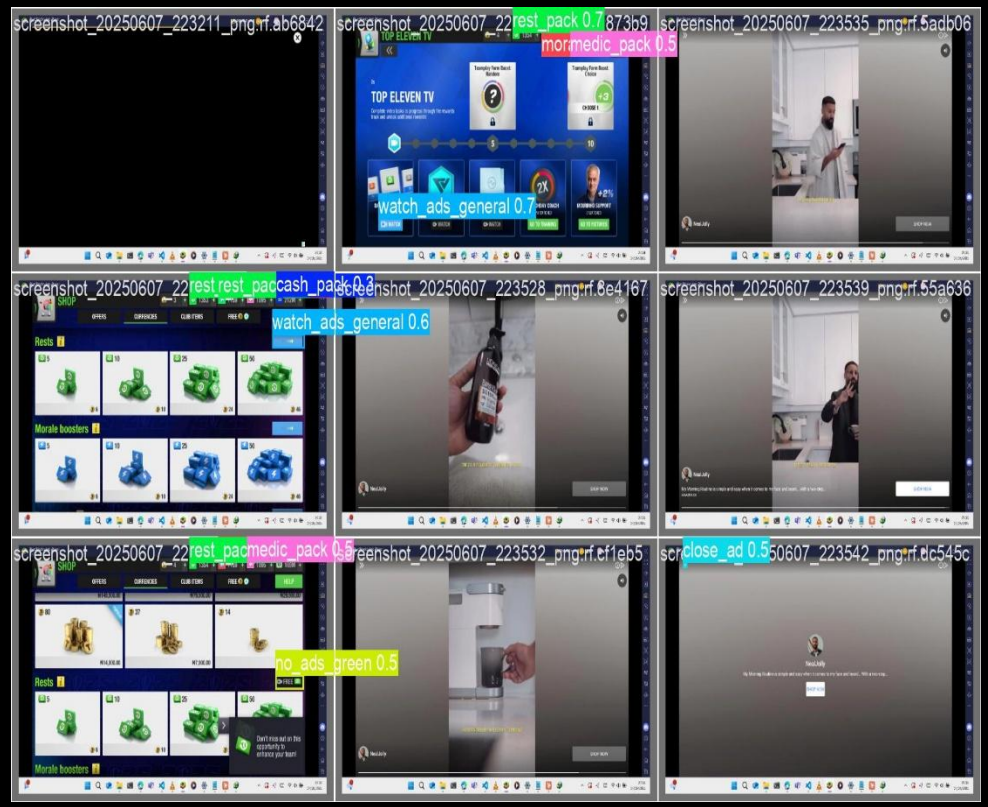
# Experiments and Results

```
from IPython.display import Image as IPyImage

IPyImage(filename=f'{HOME}/runs/detect/train/confusion_matrix.png', widt
```




Confusion Matrix




Confidence against test images

- Dataset: 50 annotated screenshots, 5+ object classes
- Training: 10 epochs, YOLO11s model (9.4M parameters)
- Performance: Overall mAP 0.651, best class (skip_ad) achieved 1.0 mAP
- Real-world Success: 85% automation success rate in live gameplay

# Discussions and Limitations

1. Model Generalization: Limited to current Top Eleven UI version - performance degrades with game updates or interface changes
2. Detection Accuracy: 10-15% false positive rate for similar UI elements, particularly challenging with dynamic ad content variations
3. System Dependencies: Windows-only implementation requiring specific BlueStacks configuration and fixed screen resolution
4. Ethical Considerations: Potential violation of game terms of service raises questions about automated vs. manual gameplay fairness

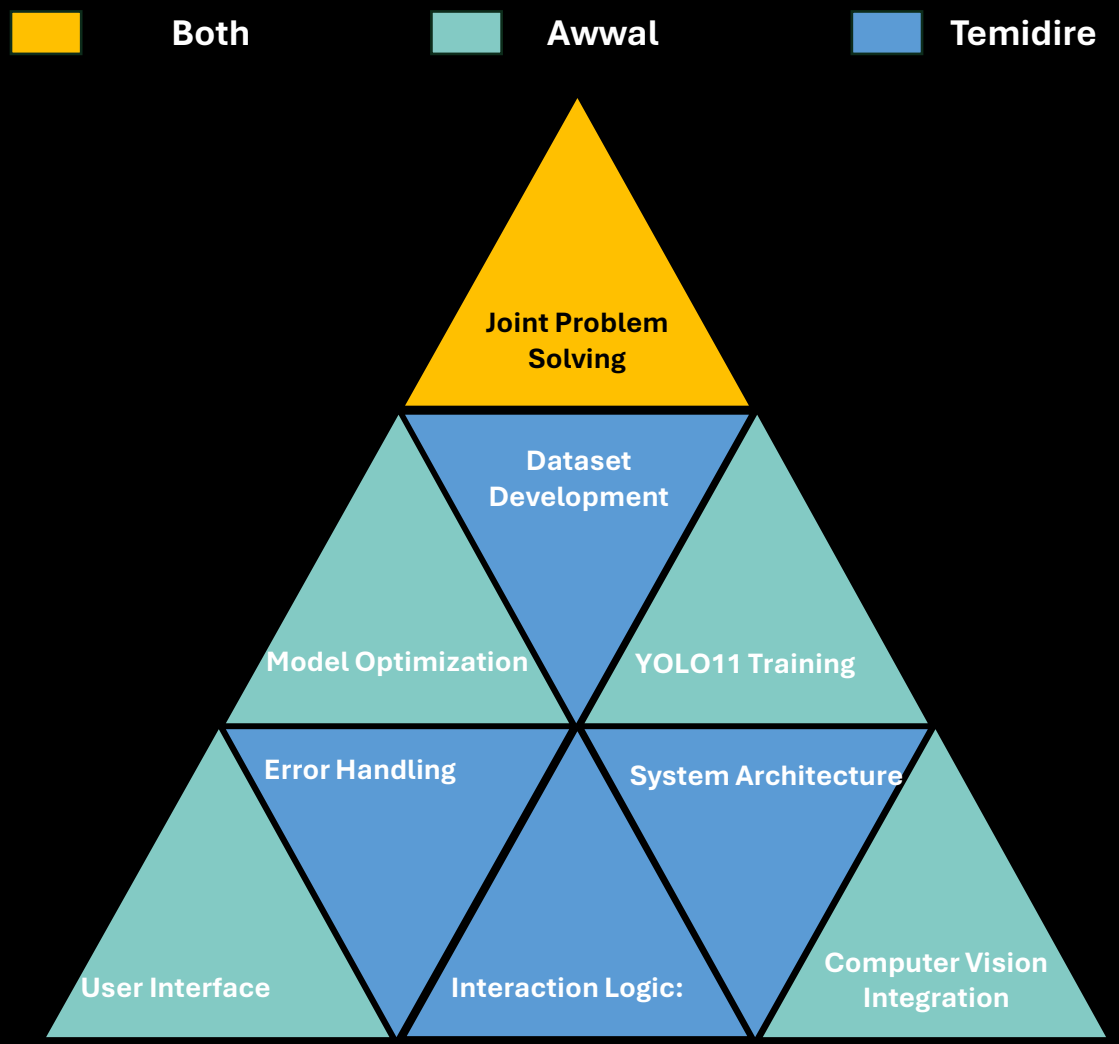
# Conclusion and Future works

Adding more object classes to be detected.

Adding more functionality to the Bot (player training, transfer market)

Training more models for  different screens sizes and resolutions (1920 x 1080, 2560 × 1664, 2880 × 1800 )

# Team Contribution



# References


1. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788). IEEE. https://doi.org/10.1109/CVPR.2016.912.
2. Chen, X., Wang, S., Fu, B., Long, M., & Wang, J. (2019). Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. In Advances in neural information processing systems (pp. 1906-1916). Neural Information Processing Systems Foundation. https://doi.org/10.5555/3454287.34544573.
3. Jocher, G., Chaurasia, A., & Qiu, J. (2023). Ultralytics YOLOv8: A new real-time object detection and image segmentation model. Ultralytics. Retrieved from https://github.com/ultralytics/ultralytics