SMART INDIA HACKATHON 2024

SMART INDIA HACKATHON 2024

- **Problem Statement ID – 1671**
- **Problem Statement Title- Develop a functional solution that demonstrates the face liveness detection**
- **Theme- Smart Automation**
- **PS Category- Software**
- **Team ID- 24221**
- **Team Name - Neural NIti**

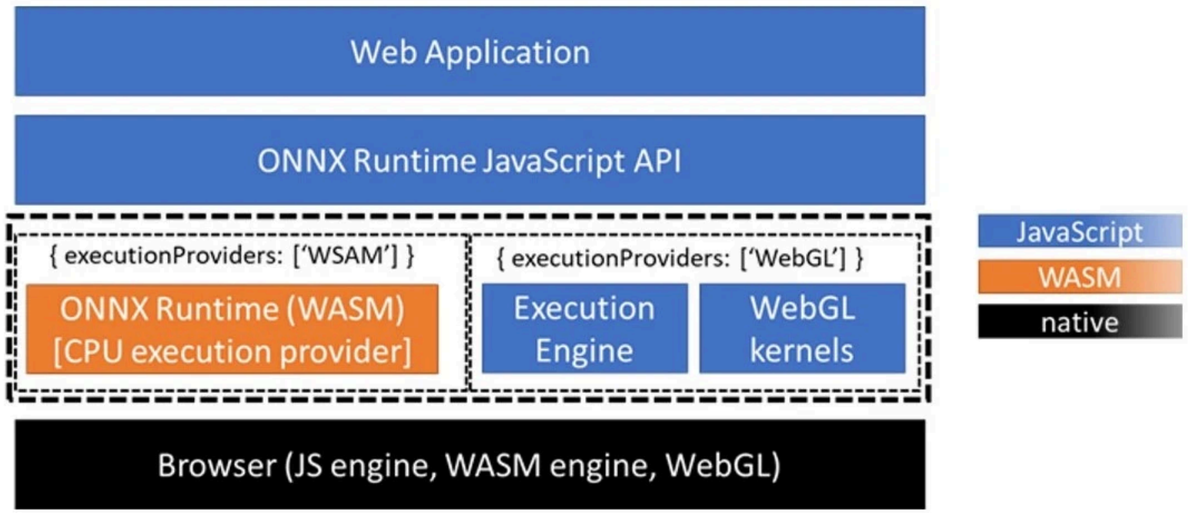
Neural NIti

Hybrid Face Liveness Detection Using WebAssembly for Real-Time Browser-Based Security

SMART INDIA HACKATHON 2024

1. Detailed Explanation

- Browser-based implementation using ONNX or TensorFlow.js, leveraging **WebAssembly (WASM)** for efficient execution.
- Cross-browser compatibility **(Chrome, Firefox, Edge)** ensured through WASM integration.
- Compact **2 MB model size** with rapid **300 ms inference time**, optimized for WASM execution.
- The model detects a wide range of **advanced spoofing techniques**, including relay, mask, print, makeup, 2D-to-3D, glasses, and light-based attacks, offering state-of-the-art anti-spoofing capabilities.



2. How It Addresses the Problem

- Comprehensive Verification: Combines active and passive methods for **robust liveness detection**, preventing photo/video spoofing.
- Cross-browser compatibility: Ensures widespread accessibility with cross-browser compatibility via **WASM Deployment on the Client side.**
- Less Bandwidth Consumption: Quick loading on **3G/4G networks due to small model size of 2 MB** and efficient WASM implementation.
- Aligns with **UIDAI's requirement** for same-device processing and security.

3. Innovation and Uniqueness

- **Hybrid Model:** Integrates both **active and passive** detection for enhanced security.
- **Fast inference and load time:** Leveraging WebAssembly for high-performance, browser-based liveness detection provides **faster execution and lower memory** usage compared to traditional JavaScript-based solutions.

Neural NIti

TECHNICAL APPROACH

SMART INDIA HACKATHON 2024

Technologies to be Used

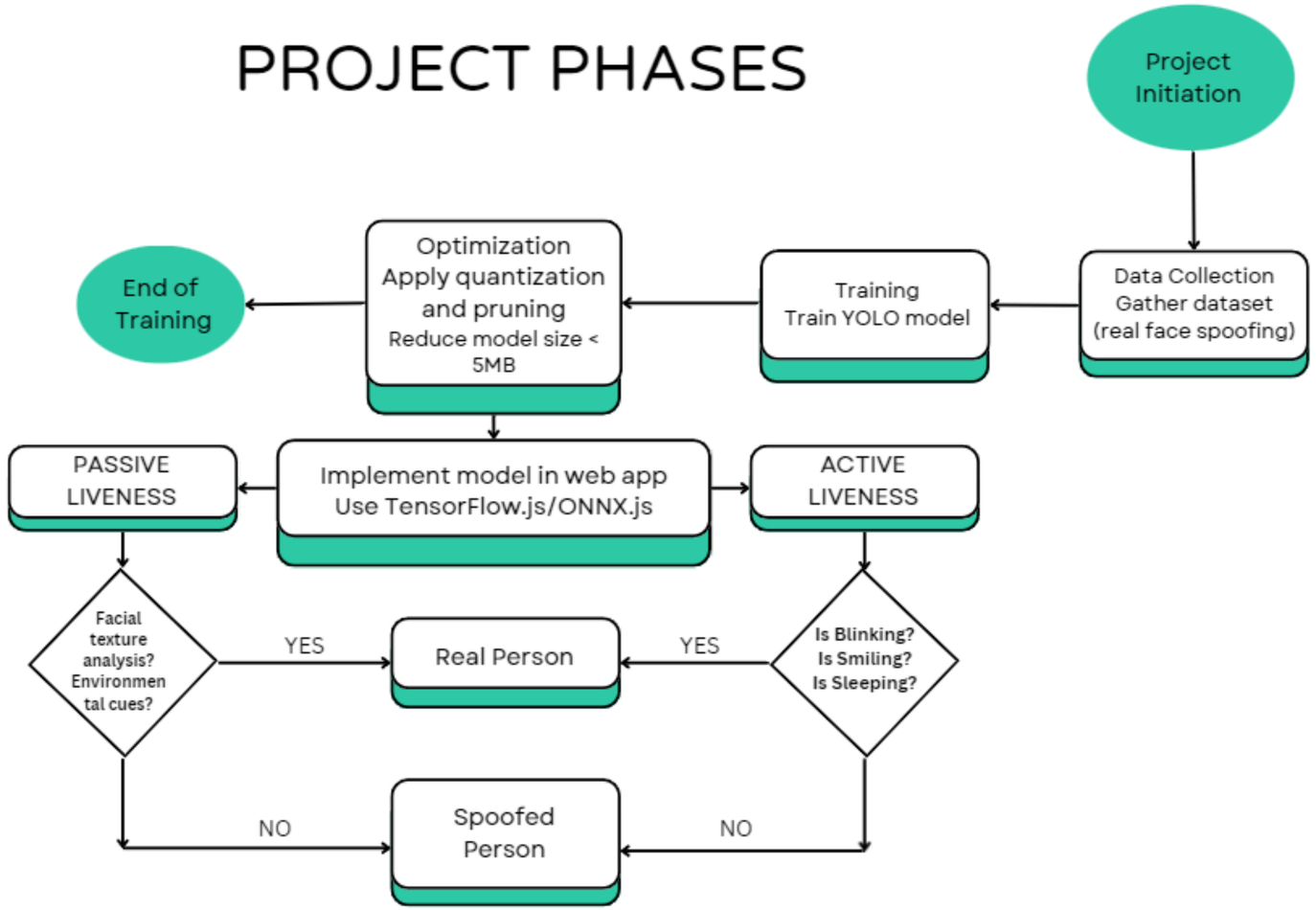
1. **Programming Languages**
   - JavaScript
   - Python, C and C++
   - HTML

2. **Frameworks and Libraries**
   - YOLOv8n (nano)
   - WEBAssembly
   - OpenCV

3. **Hardware**
   - Desktop, mobile, or tablet with a camera

Methodology and Process for Implementation

- Model Training and Optimization: Train and optimize customize YOLO v8 on real face and spoofing data.
- Browser Integration: Implement the model with active and passive liveness in a web app.
- Performance Optimization: Use WebAssembly to ensure 500ms execution time.
- Deployment and Monitoring



Neural NIti

FEASIBILITY AND VIABILITY

SMART INDIA HACKATHON 2024

Analysis of the Feasibility of the Idea

- WebAssembly's wide browser support and near-native performance enable the 300ms inference time across major platforms.
- The 2MB model size, optimized for WASM, fits well within the 5MB limit while maintaining effectiveness.
- WASM's capabilities support the integration of both passive and active liveness detection in a browser context.

Potential Challenges and Risks

- Performance Variability: Inconsistent browser performance across devices may cause detection delays.
- Model Accuracy: False positives or negatives in liveness detection could reduce system reliability.
- Python and Library Incompatibility: Different versions of Python libraries (TensorFlow, OpenCV) may cause incompatibility issues.

Strategies for Overcoming These Challenges

- Optimize the model and test it on various browsers, devices, and operating systems.
- Improve the dataset and use transfer learning to fine-tune model accuracy for better spoof detection.
- Use virtual environments for version control, test for compatibility regularly, and utilize Docker for consistent development and deployment environments. Maintain dependency management with requirements.txt or environment.yml.

Neural NIti

IMPACT AND BENEFITS

SMART INDIA HACKATHON 2024

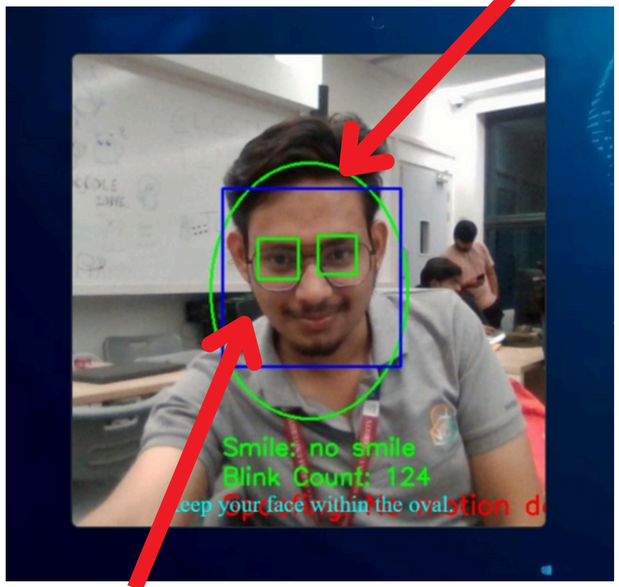
Potential Impact on the Target Audience

- Aadhaar number holders benefit from enhanced security through robust face liveness detection, preventing identity theft and fraud.
- A seamless browser-based solution allows users to authenticate themselves without additional software or devices, improving the overall user experience.

Benefits of the Solution

- **Social Benefits:** Strengthens identity verification processes, reduces fraud, and improves trust in Aadhaar-based authentication services.
- **Economic Benefits:** Reduces the cost of fraudulent activities, saving organizations and users money. Streamlined verification processes can lead to more efficient service delivery.
- **Environmental Benefits:** By enabling remote, secure identity verification, the solution can reduce the need for physical documents, lowering paper consumption and the environmental footprint of the authentication process.

The model is analyzing the background and telling the person to fit into the oval for a passive detection.



The model is analyzing key interactive features like blinking, smiling or drowsiness for an active detection.

Note: Image used in this Slide is a Snap of Successful Implementation of Project Developed in Internal Hackathon 2024

Neural NIti

RESEARCH AND REFERENCES

SMART INDIA HACKATHON 2024

- TensorFlow. “TensorFlow.js: Machine Learning in the Browser.” Accessed September 6, 2024. https://www.tensorflow.org/js.
- OpenCV. “OpenCV.js Documentation.” Accessed September 6, 2024. https://docs.opencv.org/4.x/d5/d10/tutorial_js_root.html.
- Computer Vision Zone. “Code and Files for YOLO Face Detection.” Accessed September 6, 2024. https://www.computervision.zone/lessons/code-and-files-31/.
- Elyha7. “YOLO Face Detection.” GitHub. Accessed September 6, 2024. https://github.com/elyha7/yoloface.
- IEEE Xplore. “Face Liveness Detection Technology Overview.” Accessed September 6, 2024. https://ieeexplore.ieee.org/document/873873
- Microsoft Open Source. (2021). ONNX Runtime Web: Running Your Machine Learning Model in Browser. Retrieved from Microsoft Open Source Blog.