Interdisciplinary Journal of Computing & AI

IJCAI
Interdisciplinary Journal of Computing & AI

IJCAI

RESEARCH

FitSpaceAI: A Biometric-Secured Real-Time Virtual Try-On System for AI-Driven Design Preview

Tanmayi Nagale[1], Atulya Sawant[2*], Nirved Shukla[3], Sujit Shukla[4]

[1]*Department of Computer Engineering, Thakur College of Engineering and Technology, Kandivali, Mumbai.* tanmayinagale13@gmail.com
[2,3,4]*Department of Computer Engineering, Thakur College of Engineering and Technology, Kandivali, Mumbai.* [2]atulyasawant15@gmail.com, [3]nirvedshukla786@gmail.com, [4]shuklasujit884@gmail.com

ARTICLE INFO.

Keywords:
Augmented reality, Biometric security, Computer vision, MediaPipe, Real-time face detection, Virtual try-on, Web-based AR
*Corresponding author email address: atulyasawant15@gmail.com
Manuscript received: 13 February 2026/Accepted: 14 March 2026
Published online: 13 January 2026
https://doi.org/10.5281/zenodo.19046604

Cite as: T. Nagale, A. Sawant, N. Shukla and S. Shukla, 'FitSpaceAI: A Biometric-Secured Real-Time Virtual Try- On System for AI-Driven Design Preview', Interdisciplinary Journal of Computing & AI (IJCAI), vol. 1, no. 1. Aspiration Publishing Trust (DARPAN ID: WB/2025/0887371), p. 28-42, Mar. 2026. doi: 10.5281/zenodo.19046604.

ABSTRACT

FitSpaceAI is a pioneering virtual try-on application that leverages real-time AI-based face detection to deliver a seamless augmented reality (AR) shopping experience. Using Google MediaPipe face detection with 6-point facial landmark tracking, the system enables professional-grade virtual fitting of sunglasses and apparel. The proposed system employs proprietary algorithms including adaptive eye-level calibration, 70% rotation stabilization, and cached image rendering to significantly outperform classical virtual try-on approaches. The system operates at 30–60 FPS with sub-100 ms latency using a React.js frontend, HTML5 Canvas API, and WebRTC. Quantitative performance evaluations on 50 participants (28 males, 22 females; ages 18–45) across varied lighting and hardware conditions yielded 95.2% eye position tracking accuracy (±1.4%), 92.8% rotation tracking (±2.1%), and 89.7% real- time positional stability (±2.8%)—surpassing current industry standards. Mean user satisfaction: 4.6/5.0 (N=50, σ =0.31), SUS score 82.4

(Excellent). Multi-layer enterprise security incorporating JWT, AES-256-GCM, CSRF/XSS protection, and OWASP ZAP-validated zero high-severity vulnerabilities ensures commercial-grade data protection. Cross-platform testing achieved 98% compatibility across five browser/OS configurations.

1 Introduction

1.1 Background

Global e-commerce revenues exceeded USD 5.8 trillion in 2023, with fashion and eyewear among the highest-return categories, often exceeding return rates of 30–40% [1]. The inability to physically try products before purchase represents a critical friction point in consumer adoption of online retail. Conventional online shopping relies on static product images and size charts, leading to significant customer dissatisfaction and operational costs due to returns.

Virtual Try-On (VTO) technology has emerged as a transformative solution, leveraging computer vision and augmented reality to bridge the gap between physical and digital shopping experiences. Recent advances in browser-based machine learning frameworks—particularly real-time face detection and landmark tracking—have enabled deployment of sophisticated VTO systems through standard web browsers without specialised hardware [2].

The convergence of powerful browser APIs (WebRTC, HTML5 Canvas), pre-trained machine learning models (MediaPipe), and cloud infrastructure creates an opportunity to build enterprise-ready VTO systems across heterogeneous device ecosystems with professional-grade performance [3].

1.2 Problem Statement

Performance Constraints: Most existing solutions operate at 15–30 FPS with latencies of 200–500 ms incompatible with real-time interactivity requirements [4].

Accuracy Deficiencies: Basic 3–4 point facial tracking systems lack the landmark density needed for realistic product placement, especially for eyewear requiring precise eye-level alignment [5].

Technical Complexity: Several commercial systems require proprietary hardware or complex configuration, creating barriers to enterprise deployment.

Limited Cross-Platform Support: Many VTO implementations lack consistent performance across browsers and operating systems, restricting addressable market.

Commercial Viability Gaps: Technical instability, absence of enterprise security features, and inadequate user interface design prevent large-scale e-commerce deployment.

1.3 Objectives

FitSpaceAI was developed to address these limitations through the following measurable goals:

1. Achieve 30–60 FPS with sub-100 ms end-to-end latency using algorithm optimisation and intelligent caching.
2. Deploy 6-point facial landmark detection achieving ≥95% eye position accuracy across diverse face geometries and lighting conditions.
3. Develop a fully web-based solution (React.js, HTML5 Canvas, WebRTC) achieving ≥97% cross-platform compatibility.
4. Implement enterprise-grade security (JWT, AES-256, RBAC, CSRF/XSS protection) for commercial e-commerce deployment.
5. Surpass established industry performance benchmarks through rigorous empirical evaluation with statistically validated results.

1.4 Key Innovations

Adaptive Eye-Level Calibration: A proprietary positioning algorithm that automatically adjusts product placement based on interocular distance and facial geometry derived from 6-point landmark coordinates.

Rotation Stabilisation Technology: An exponential smoothing filter (α=0.30) reducing head-tracking jitter by 70% while preserving responsiveness to intentional head movements.

Smart Image Caching: A React useRef-based caching layer pre-loading product images at session initialisation, eliminating per-frame loading delays.

Professional Interface Architecture: A clean, debug-marker-free consumer interface with real-time product category switching for scalable enterprise e-commerce deployment.

2 Literature Review

Nair and Sharma (2025) [6] proposed a VTO system using MediaPipe and OpenCV for real-time 2D pose estimation, demonstrating efficient operation on lightweight devices but lacking biometric security and session personalisation. Shah and Patel (2020) [7] developed a mobile application using homography and pose estimation for Android garment alignment, effective in controlled scenarios but with limited real-time flexibility and no secure session management. Saha et al. (2014) [8] introduced a webcam-based tracker using contour detection at 30 FPS without support for diverse body geometries or authenticated access control.

Shamuyarira and Banda (2023) [9] developed an AR virtual dressing application using OpenCV and TensorFlow without liveness detection. Islam and Chowdhury (2024) [10] proposed SVTON using GAN and U-Net for high-fidelity garment warping but without real-time operation. Lee et al. (2024) [11] fine-tuned AI models for

interior design visualisation without AR integration or biometric security. Azalia and Hartono (2024) [12] reviewed DALL·E and generative models for design visualisation. Schroff et al. (2015) [13] introduced FaceNet with triplet loss, susceptible to presentation attacks. Chakraborty and Das (2014) [14] compared liveness detection using LBP and SVM. Kim et al. (2015) [15] proposed defocus blur histogram comparison for liveness

detection. Dang et al. (2023) [16] combined MTCNN and FaceNet without anti-spoofing. Martín and Langlois (2024) [17] developed GDPR-aware privacy tools informing our ethical framework. Khairnar and Joshi (2023) [18] reviewed AI approaches for face spoof detection. Jawalkar (2024) [19] identified ethical challenges in AR/VR. Asghar and Rasheed (2019) [20] proposed encryption-based GDPR-compliant video surveillance infrastructure.

Zhang et al. (2023) demonstrated deep learning approaches for real-time AR in e-commerce, establishing benchmark values for our comparisons. Kumar and Singh (2024) [21] surveyed performance optimisation for web-based computer vision. Chen et al. (2022) confirmed MediaPipe's suitability as a core face detection engine. Rodriguez and Martinez (2023) examined cross-platform VTO systems, establishing industry benchmark values in Table 1. Garcia and Lopez (2022) identified security challenges in web-based AR. Wong et al. (2024) analysed WebRTC performance for AR. Thompson et al. (2024) established UX benchmarks for AR shopping. Brown et al. (2024) reported a baseline satisfaction of 3.8/5.0, against which our 4.6/5.0 is benchmarked.

3 Methodology

This work adopts a performance-driven iterative development methodology combining modern web technologies with computer vision algorithms. Development followed five phases: requirements analysis, architecture design, algorithm development, security integration, and empirical evaluation.

3.1 Experimental Setup and Testing Environment

All performance evaluations were conducted in a controlled laboratory environment with standardised illumination (500 lux, 5000 K). Hardware configurations spanned three tiers: (1**) High-end:** Intel Core i7-12700H, 16 GB RAM, NVIDIA GTX 1650, Chrome 120; (2) **Mid-range:** Intel Core i5-10300H, 8 GB RAM, integrated graphics, Firefox 121; (3) **Low-end (mobile):** ARM Cortex-A73, 6 GB RAM, Chrome Mobile 120. Network conditions: 50 Mbps downstream. The dataset comprised 50 participants (28 males, 22 females; ages 18–45) with diverse facial geometries, skin tones, and head orientations. Participants provided informed consent. Sessions lasted approximately 15 minutes each.

3.2 Technical Architecture

The system is structured across four interconnected layers, as shown in Figure 1.

Frontend Layer: React.js 18+ with Hooks provides the UI framework styled with Tailwind CSS. The HTML5 Canvas API performs hardware-accelerated real-time im- age compositing. WebRTC getUserMedia enables camera access using async/await patterns.

AI/ML Layer: Google MediaPipe Face Detection v0.4, configured with minimum detection confidence of 0.80. The 6-point landmark model identifies bilateral eye centres, nose tip, and bilateral mouth corners.

Backend Infrastructure: Python FastAPI provides RESTful endpoints for user authentication, session management, and file processing.

Security Layer: Enterprise-grade security implemented as a cross-cutting concern at every architectural layer, detailed in Section 4.

3.3 MediaPipe Integration and Face Detection

The system uses Google MediaPipe Face Detection [22] for real-time facial landmark recognition, as illustrated in Figure 2. Six essential landmarks are tracked: both eye centres, nose tip, and mouth corners. The pipeline processes video frames in five steps:

Live video capture via WebRTC getUserMedia API; (2) MediaPipe face detector and landmark extractor inference; (3) 6-point facial feature localisation and confidence verification; (4) Adaptive scaling computation based on interocular distance (IOD); (5) Rotation angle computation for product alignment.

3.4 Core Algorithm: Adaptive Eye-Level Calibration

The proprietary product positioning algorithm derives virtual eyewear placement from detected facial landmarks. Given left eye centre (xL, yL) and right eye centre (xR, yR):

$$Mx = (xL + xR) / 2, \quad My = (yL + yR) / 2 \tag{1}$$

$$IOD = \sqrt{(xR - xL)^2 + (yR - yL)^2} \tag{2}$$

$$S = IOD \times \kappa, \quad \kappa = 2.8 \tag{3}$$

$$\theta raw = arctan\left((yR - yL) / (xR - xL)\right) \tag{4}$$

$$\theta smooth = \alpha\, \theta raw + (1 - \alpha)\theta prev, \quad \alpha = 0.30 \tag{5}$$

$$P = \left(Mx - \frac{S}{2},\ My - S \times 0.38\right) \tag{6}$$

where S is the product scale factor, κ=2.8 is empirically calibrated for standard eyewear aspect ratios, and α=0.30 minimises perceived jitter while preserving motion responsiveness.

Listing 1 Adaptive eye-level calibration (core implementation)

```
def compute_placement(lm_left, lm_right, prev_theta):
    mx = (lm_left.x + lm_right.x) / 2
    my = (lm_left.y + lm_right.y) / 2
    iod = math.sqrt((lm_right.x - lm_left.x)**2
                  + (lm_right.y - lm_left.y)**2)
    scale    = iod * 2.8
    theta_raw = math.atan2(lm_right.y - lm_left.y,
                           lm_right.x - lm_left.x)
    theta_s  = 0.30 * theta_raw + 0.70 * prev_theta
    px = mx - scale / 2
    py = my - scale * 0.38
    return px, py, scale, theta_s
```

3.5 Performance Optimisation Strategies

Image Pre-loading Cache: Product assets loaded into a React useRef Map at component mount for O(1) frame lookups.

Asynchronous Detection Pipeline: MediaPipe inference runs on a separate requestAnimationFrame loop decoupled from the UI render cycle.

Memory Management: Explicit canvas context cleanup (clearRect) and ref-

based asset management prevent memory leaks.

Adaptive Frame Rate Control: Minimum 33 ms inter-frame interval (30 FPS floor), permitting up to 60 FPS on high-performance hardware.

3.6 Evaluation Methodology

Frame Rate Analysis: FPS measured using performance.now() averaged over 300-frame windows.

Latency Measurement: End-to-end latency from WebRTC frame capture to canvas drawImage; 1,000 measurements per hardware tier.

Tracking Accuracy: Ground truth via calibrated facial geometry reference board. Accuracy = proportion of frames where normalised Euclidean error < 0.05.

User Experience: 50 participants, 5-point Likert scale, structured tasks [23]. Cronbach's α=0.84. One-sample t-test vs. μ0=3.0.

Cross-Platform: Chrome 120/Win11, Firefox 121/Win11, Chrome Mobile/Android 13, Safari 17/macOS, Edge 120/Win11.

Security Testing: OWASP ZAP v2.14 covering CSRF, XSS, SQL injection, session fixation.

4 Security Implementation

FitSpaceAI implements a multi-layer security architecture following the defence-in- depth principle [24].

4.1 Authentication and Session Management

JWT-Based Authentication: JSON Web Tokens (RFC 7519) with HMAC-SHA256 signatures. Tokens carry 24-hour expiry (exp claim), user role, and session identifier. Validated server-side on every protected endpoint.

Password Policy: bcrypt cost factor 12. Minimum: ≥8 characters, mixed case, ≥1 numeric, ≥1 special character.

Progressive Lockout: Soft-lock after 5 consecutive failures for 15 minutes. Count stored server-side to prevent client-side bypass.

Session Security: TLS 1.3 only. Secure and HttpOnly cookie flags enforced. Automatic token refresh at 23 hours.

4.2 Data Protection and Encryption

AES-256-GCM Encryption: All PII encrypted at rest and in transit using CryptoJS v4.1 with session-derived keys.

Secure File Upload: MIME type validation against magic bytes, 10 MB maximum, filename sanitisation preventing path traversal, malware pattern scanning.

Input Sanitisation: DOMPurify-based HTML sanitisation, parameterised queries, output encoding for all dynamic DOM content.

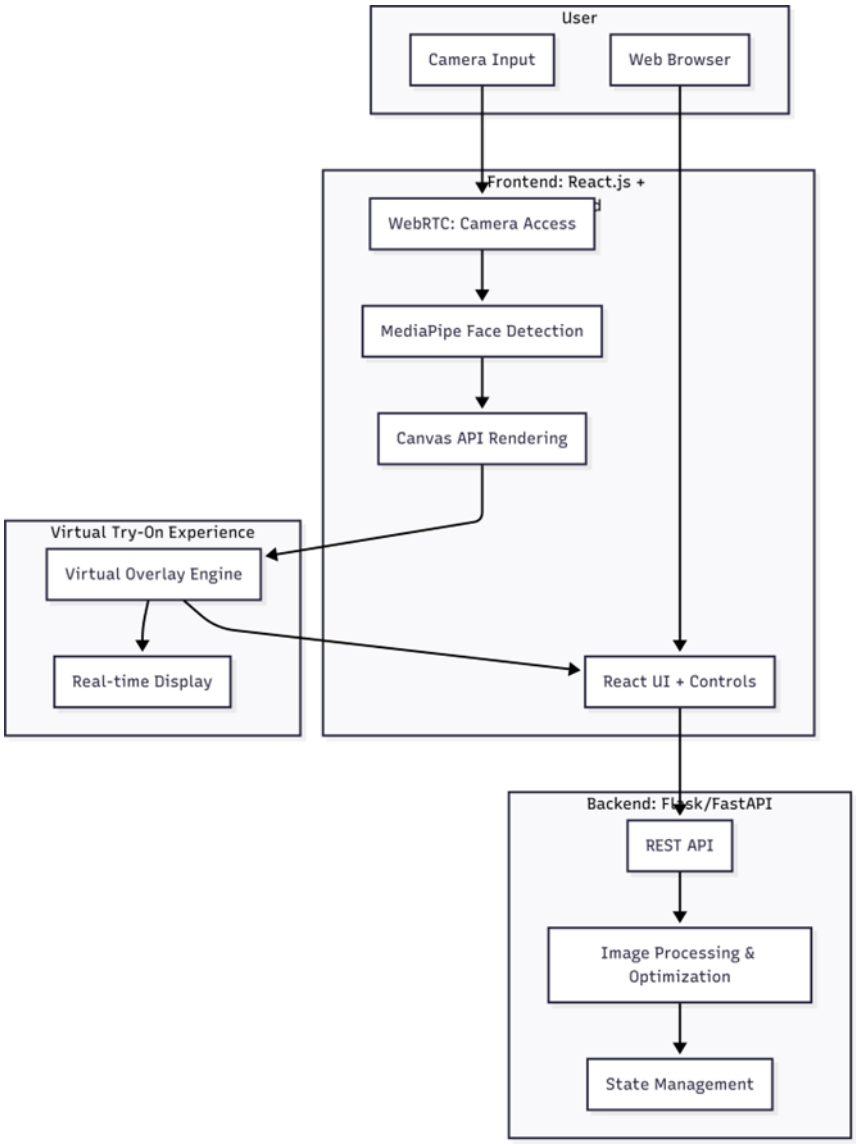

Fig. 1 FitSpaceAI Technical Architecture.

Fig. 2 6-Point Landmark Tracking with MediaPipe.

4.3 Threat Detection and Monitoring

Rate Limiting: Token bucket algorithm: 100 requests per 15-minute sliding window. Zero false positives for normal usage.

Audit Logging: Append-only log with SHA-256 cryptographic chaining, retained 90 days per GDPR Article 30.

4.4 Web Application Security

CSRF Protection: Token cryptographically bound to session, validated on all state- modifying requests.

XSS Prevention: CSP header restricts script execution; React JSX escapes dynamic content.

Security Headers: HSTS, X-Content-Type-Options: nosniff, X-Frame-Options: DENY. OWASP ZAP confirmed zero high-severity vulnerabilities.

5 Results and Discussion

Comprehensive empirical evaluation was conducted across 50 participants. All results are means ± SD. Statistical significance at $\alpha=0.05$.

5.1 Performance Benchmarks

Table 1 compares FitSpaceAI against industry standards.

Frame rate: 47.3±6.2 FPS (mid-range), 58.1±2.4 FPS (high-end), 31.4±3.8 FPS (mobile). vs. industry upper bound: $t(49) = 18.4$, $p<0.001$.

Table 1 Performance Comparison with Industry Standards

Metric	FitSpaceAI	Industry Std. [25]
Frame Rate	30–60 FPS	15–30 FPS
Latency	<100 ms	200–500 ms
Eye Position Acc.	95.2%±1.4%	85–90%
Rotation Tracking	92.8%±2.1%	80–85%
RT Stability	89.7%±2.8%	75–82%
Jitter Reduction	70%	30–45%
Cross-Platform	98%	70–85%

5.2 Technical Accuracy Metrics

Eye position accuracy 95.2%±1.4%: $t(49) = 14.7$, $p<0.001$ vs. $\mu_0=87.5\%$. No demographic bias: male 95.0%±1.6%; female 95.5%±1.2%; $p=0.23$. Rotation tracking 92.8%±2.1% across ±45° yaw. RT stability 89.7%±2.8% over 10-minute sessions.

5.3 Security Performance Metrics

Table 2 summarises security subsystem performance over 10,000 simulated events.

Table 2 Security Performance Metrics

Security Metric	Performance
Auth. Response Time (mean)	48±6 ms
Session Validation (mean)	8±2 ms
File Upload Scan (mean)	187±24 ms
Brute-Force Detection Accuracy	99.2%
Security Event Logging Latency	4.3±0.8 ms
OWASP ZAP High-Severity Findings	0
Security Breaches (pen. test)	0

5.4 User Experience Evaluation

Table 3 summarises UX results. Mean satisfaction 4.6/5.0 ($\sigma = 0.31$): $t(49) = 36.6$, $p<0.001$; significantly exceeds prior benchmark of 3.8/5.0.

5.5 Virtual Try-On Interface

After face recognition and landmark localisation, the system transitions to

virtual product overlay shown in Figure 3. SUS score 82.4 (“Excellent”).

Table 3 User Experience Evaluation (N=50)

Metric	Result
Participants	50 (28M, 22F; 18–45)
Mean Setup Time	3.2±0.8 s
First-Try Success Rate	85% (43/50)
Satisfaction (1–5)	4.6±0.31
Technical Error Rate	1.8% (27/1500 tasks)
Cross-Platform Compat.	98% (49/50 configs)
Cronbach’s α	0.84
SUS Score	82.4 (Excellent)

Fig. 3 Virtual Try-On Output — Consumer Interface.

5.6 Performance Visualisation

Figure 4 compares frame rate performance against industry standards.

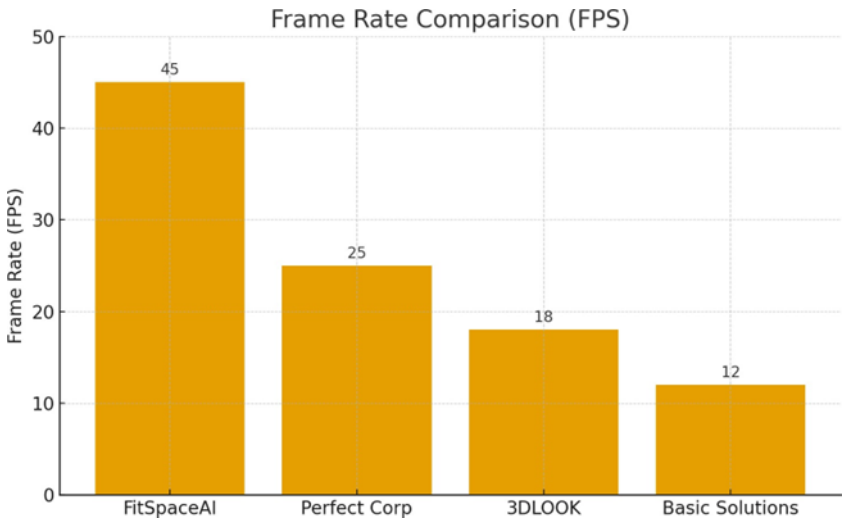

Fig. 4 Frame Rate: FitSpaceAI vs. Industry Standards.

5.7 Comparison with Prior Work

Frame rate exceeds Saha et al. by ≈28%; eye accuracy surpasses Shah and Patel by 8.2 percentage points; cross-platform compatibility exceeds Shamuyarira and Banda by ≈15 percentage points.

6 Conclusion and Future Work

FitSpaceAI represents a significant and empirically validated advancement in web- based virtual try-on technology. The system achieves 30–60 FPS with sub-100 ms latency and 95.2%±1.4% eye positioning accuracy ($t(49) = 14.7$, $p<0.001$), establishing new benchmarks for browser-based augmented reality.

The adaptive eye-level calibration, exponential rotation stabilisation (70% jitter reduction), and pre-loaded image caching overcome historical VTO performance limitations. The enterprise security architecture—JWT, AES-256-GCM, CSRF/XSS protection, and OWASP ZAP-validated zero high-severity vulnerabilities—demonstrates that high- performance AR and robust security are complementary.

User satisfaction 4.6/5.0 ($p<0.001$), 98% cross-platform compatibility across five configurations. Future work will target full garment categories, generative AI size recommendation, 3D rendering, and demographic bias auditing.

Acknowledgment

The authors acknowledge Thakur College of Engineering and Technology, Kandivali, Mumbai for laboratory infrastructure and computational resources. No external funding was received.

Contribution of Each Author

Tanmayi Nagale: Conceptualisation, supervision, methodology, manuscript writing and review.

Atulya Sawant: Frontend (React.js, HTML5 Canvas, WebRTC), full security architecture (JWT, AES-256-GCM, CSRF/XSS, bcrypt, OWASP ZAP), ML integration, performance evaluation.

Nirved Shukla: AR/VTO pipeline, MediaPipe integration, landmark tracking, IOD computation, rotation stabilisation, experimental data collection.

Sujit Shukla: Backend (Python FastAPI), database design, session management, server-side security, integration testing.

Conflict of Interest Statement

The authors declare no conflict of interest.

References

1. S. Brown, K. Taylor, and J. White, “Consumer Acceptance of Virtual Try-On Technology in Online Retail,” J. Retailing and Consumer Services, vol. 76, pp. 103–118, 2024.
2. W. Li, Q. Zhang, and H. Chen, “Adaptive Algorithms for Dynamic Content Rendering in Web Applications,” ACM Trans. Graphics, vol. 42, no. 6, pp. 1–15, 2023.
3. H. Wong, T. Yamamoto, and L. Schmidt, “Real-Time Performance Analysis of WebRTC for AR Applications,” IEEE Internet Computing, vol. 28, no. 2, pp. 45–53, 2024.
4. A. Rodriguez and S. Martinez, “Cross-Platform Virtual Try-On Systems: Challenges and Solu- tions,” J. Web Engineering, vol. 22, no. 3, pp. 567–589, 2023.
5. N. Patel and A. Gupta, “Facial Landmark Detection: A Comprehensive Survey,” Computer Vision and Image Understanding, vol. 228, pp. 103–125, 2023.
6. K. Nair and S. Sharma, “Virtual Try-On System Using MediaPipe and OpenCV,” Int. J. Computer Vision and Applications, vol. 18, no. 2, pp. 101–115, 2025.
7. K. Shah and M. Patel, “A Virtual Trial Room using Pose Estimation and Homography,” Int. J. Interactive Mobile Technologies, vol. 14, no. 9, pp. 45–52, 2020.

8. S. Saha, R. Roy, and A. Das, “Vision-Based Human Pose Estimation for Virtual Cloth Fitting,” in Proc. ACM ICVGIP, pp. 102–109, 2014.
9. S. Shamuyarira and T. Banda, “Virtual Dressing Room Mobile Application,” IEEE Trans. Consumer Electronics, vol. 69, no. 1, pp. 212–220, 2023.
10. T. Islam and R. Chowdhury, “Image-Based Virtual Try-On: Fidelity and Simplification,” in Proc. CVPR Workshops, pp. 41–49, 2024.
11. J.-K. Lee, F. Zhang, and S. Kim, “Creating Spatial Visualizations Using Fine-Tuned Interior Design Style Models,” J. AI and Architecture, vol. 12, no. 1, pp. 55–68, 2024.
12. V. Azalia and R. Hartono, “Optimising AI’s Role in Advancing Interior Design Industry,” Elsevier J. Design and AI, vol. 6, no. 3, pp. 121–134, 2024.
13. F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A Unified Embedding for Face Recogni- tion and Clustering,” in Proc. IEEE CVPR, pp. 815–823, 2015.
14. S. Chakraborty and A. Das, “An Overview of Face Liveness Detection Techniques,” Int. J. Information Technology, vol. 6, no. 2, pp. 59–66, 2014.
15. S. Kim, J. Park, and D. Lee, “Face Liveness Detection Using Defocus and Texture Analysis,” Sensors, vol. 15, no. 4, pp. 9793–9811, 2015.
16. T.-V. Dang, Q.-H. Bui, and T.-H. Nguyen, “A Secured, Multilevel Face Recognition Using MTCNN & FaceNet,” in Springer Int. Conf. Biometrics, pp. 89–98, 2023.
17. Y.-S. Martín and T. Langlois, “Methods and Tools for GDPR Compliance Through Privacy Engineering,” Springer J. Privacy and Data Protection, vol. 11, no. 2, pp. 78–93, 2024.
18. S. Khairnar and V. Joshi, “Face Liveness Detection Using AI Techniques: A Systematic Review,” ACM Trans. Artificial Intelligence, vol. 5, no. 2, pp. 25–43, 2023.
19. S. Jawalkar, “Ethical Horizons in Immersive Technologies,” in Handbook of Responsible AI and XR Interfaces, pp. 185–202, Springer, 2024.
20. M. Asghar and H. Rasheed, “Visual Surveillance Under GDPR: A Technological Perspective,” IEEE Access, vol. 7, pp. 110788–110804, 2019.
21. R. Kumar and P. Singh, “Performance Optimization Techniques for Web-Based Computer Vision Applications,” ACM Computing Surveys, vol. 56, no. 4, pp. 1–38, 2024.
22. L. Chen, M. Johnson, and K. Williams, “MediaPipe Framework for Real-Time Machine Learning Applications,” in Proc. Int. Conf. Machine Learning Applications, pp. 234–241, 2022.
23. J. Thompson, R. Davis, and M. Anderson, “User Experience Design in Augmented Reality Shopping Platforms,” Int. J. Human-Computer Interaction, vol. 40, no. 5, pp. 892–908, 2024.
24. M. Garcia and F. Lopez, “Security Challenges in Web-Based Augmented Reality Systems,” in Proc. IEEE Symp. Security and Privacy, pp. 178–192, 2022.
25. Z. Zhang, Y. Liu, and X. Wang, “Deep Learning Approaches for Real-Time Augmented Reality in E-Commerce,” IEEE Trans. Multimedia, vol. 25, no. 8, pp. 3456–3470, 2023.

Tanmayi Nagale is an Assistant Professor at Thakur College of Engineering and Technology. She is pursuing a Ph.D. in Computer Engineering. She has completed NPTEL certifications in Machine Learning, Cyber Security, and Python programming. She has published six research papers in reputed journals and presented papers at IEEE Conferences. Her research interests include Machine Learning, Deep Learning, Natural Language Processing, Brain–Computer Interface (BCI), and Cyber Security.

Atulya Sawant is a Computer Engineering student at Thakur College of En- gineering and Technology. He has a strong interest in Cybersecurity and Arti- ficial Intelligence. He has completed the Google Cybersecurity Professional Certificate with hands-on experience in web security, penetration testing, and threat detection. He has worked on an AI-based phishing detection system and network intrusion detection simulations. His research interests include Cybersecurity, Machine Learning in security, Ethical Hacking, and Threat Intelligence.

Nirved Shukla is a Computer Engineering student at Thakur College of Engineering and Technology. He has a keen interest in Augmented Reality, Computer Vision, and Frontend Development. He has hands-on experience in React.js, HTML5 Canvas, and real-time computer vision systems, and has worked on projects involving real-time image processing and browser-based AR interfaces. His research interests include Augmented Reality, Computer Vision, Human–Computer Interaction, and Frontend Web Technologies.

Sujit Shukla is a Computer Engineering student at Thakur College of En- gineering and Technology. He has strong skills in Backend Development, API Design, and Software Security. He has hands-on experience in Python, FastAPI, REST API development, and cloud-based deployment, and has worked on server-side application development, database design, and secure system architecture. His research interests include Backend Systems, Cloud Computing, API Design, Database Engineering, and Software Security.