**Paper 5: “AI-Based Proctoring System for Online Tests” - IEEE**

**Key Contributions of the Paper:**

* The paper presents an AI-based proctoring system that utilizes LBPH for face detection and YOLOv3 for object detection. The two methods are employed to effectively authenticate student identity, detect multiple faces, and monitor head movement during exams.
* It has a mechanism to report suspicious activities such as misuse of mobile phones, switching of tabs, and other irregularities during the examination by taking occasional snapshots in between along the course of the examination process.

**Limitations:**

* The system is reliant on intermittent snapshots (e.g., 10 seconds), and they may overlook temporary or transient malpractices in between the snapshots.
* Suspicious activity is reported for further investigation instead of sending out instantaneous alerts, causing delay in intervention.
* There is no stable, two-way communication channel limiting the system to provide real-time monitoring and fast response.

**Our Solution:**

* We use WebSockets to support a stable low-latency channel of communication between the client and server. This ensures any disparity detected by LBPH or YOLOv3 is immediately communicated for the instant action to be taken.
* We leverage WebRTC to facilitate real-time continuous streaming of videos, wherein there can be unbroken live feed and live monitoring of the exam environment.
* Our approach is the combination of these real-time communication protocols with a sound backend infrastructure (Flask or FastAPI), containerization through Docker, and best-in-class load balancing through NGINX.

**Paper 6: “Research on the Development of a Proctoring System for Conducting Online Exams in Kazakhstan”**

**Key Contributions of the Paper**

* **Monitoring Browser and Keystroke Activity:**
  + The system exploits keystroke dynamics and browser monitoring to track user behaviour in real-time during web-based tests. By tracking typing behaviour, including rhythm, speed, and pressure, and browser behaviour, including tab switching, browsing URLs, and copy-paste, the system can detect inconsistencies that may suggest cheating. Machine learning models can be integrated to enable further refinement in the detection of subtle context-dependent anomalies, with a more subtle approach to differentiating between legitimate and suspicious behaviour.

* **Rule-Based Anomaly Detection:**
  + It applies a predefined set of parameters to identify broad trends of suspect behaviour, like frequent or excessive switching between tabs and prolonged periods of inactivity, that are suggestive of possible exam violations. Effective at recognizing established cheating techniques, this rule-based system is likewise extremely inflexible and doesn't account for new or innovative cheating techniques.  Incorporating adaptive anomaly detection models, including clustering or deep learning-based models, would enhance the capacity of the system to learn and adapt to new cheating techniques.

**Drawbacks**

* **Inability to Detect Multi-Device Cheating**
  + The system is limited to browser-level tracking with no external activity such as secondary device usage or messaging via third-party applications, which diminishes detection capability.

* **High False Positives in Keystroke Analysis**
  + Rule-based keystroke monitoring generates many false positives, where benign behaviour (e.g., quick typing or user disabilities) is flagged as suspicious since detection criteria are stringent.

* **Scalability and latency issues:**
  + The system suffers from inadequate real-time communication, leading to delayed alert processing, and is unable to scale, with added server load from concurrent user activity, impacting performance and response time.

**Our Solution**

* **AI-Driven Behaviour Analysis for Cross-Device Tracking:**
  + We combine OpenSeeFace and OpenCV for face recognition and dlib for tracking behaviour using unsupervised anomaly detection with cluster models to detect differences between sessions, improving detection rates and lowering false positives.

* **Improved Keystroke and Mouse Dynamics Analysis**
  + Using JavaScript and OpenCV, we utilize Reinforcement Learning to adapt to user behaviour over time, distinguishing between normal fast typists and potential cheating, optimizing dynamic detection with no predefined rules.
* **Optimizing Real-Time Communication using WebSockets and Kubernetes:**
  + WebSockets offer live notifications of abnormal activity, and Kubernetes provides automatic resource allocation for scalability. In addition, client-side processing moves keystroke analysis out of the way, reducing backend load and delay.