# MotionEye System – Technical Overview

This document provides a high-level overview of the MotionEye AutoLayout system for Raspberry Pi. It is intended to help future developers understand how the system works and how it can be deployed or maintained.

1. **System Architecture**

- The setup is built around a Raspberry Pi running MotionEye (a camera management system) and a lightweight web server (nginx). Custom frontend enhancements are delivered via a Tampermonkey user script that modifies MotionEye's web interface dynamically. Motion handles camera feeds and nginx serves static resources, including the custom user script and JSON configuration file. Tampermonkey runs inside Chromium to inject the JavaScript enhancements automatically.

2. **Core Components**

- motioneye-autolayout.user.js – A Tampermonkey userscript that customizes the MotionEye interface. It fetches camera layout data from a JSON file and dynamically adjusts the webpage (e.g., hiding headers, rearranging tiles to full screen, and auto-scrolling between cameras).
- Security-cameras.json – A JSON configuration file that defines the camera layout and display preferences.
- /var/www/html/ – The nginx web root where both the `.user.js` script and JSON config are served from.
- /home/pi/start_kiosk.sh is a bash script that is auto-run once the desktop is loaded, and starts chromium, and ensures if the process ever crashes or malfunctions it will restart and open as intended
- Chromium Browser – Loads the MotionEye web UI and automatically applies the custom behavior via the user script.

3. **Workflow**

- The Raspberry Pi boots and automatically starts MotionEye and nginx.
- Start_kiosk.sh starts Chromium, which automatically loads the motioneye URL
- Tampermonkey loads the `motioneye-autolayout.user.js` script.
- The script fetches layout data from `security-cameras.json` served by nginx.
- The page layout updates dynamically according to the JSON configuration.