

Executive Summary

This system is designed for older and mid-range vehicles that lack modern digital features, providing premium-level capabilities at a fraction of the cost of aftermarket solutions.

Background

Since Low-cost embedded platforms such as the Raspberry Pi 5 [5] now offer sufficient compute power to run advanced driver assistance features, this project aims to address gaps in existing products by building an **open-source, affordable, fully integrated solution** combining CAN bus telemetry, computer vision ADAS, AI support, and a modern web-based dashboard UI.

Cassette Futurism

Cassette Futurism [9] is a retro futuristic technological aesthetic evocative of designs from the early 1970s to mid-1990s. It is a transitional style coinciding with the emergence of digital technology in the analog world.



Figure 1. 1982 Toyota Celica Supra – An example of a car designed in the style of cassette futurism

Technology Stack

- **Hardware:** Raspberry Pi 5 (8GB), PiCAN 3 HAT, USB wide-angle camera, 10" IPS display
- **OS:** DietPi Linux, Chromium kiosk mode
- **Backend:** Python 3, Flask-SocketIO, Python-Can, OpenCV 4.8, TensorFlow Lite
- **Frontend:** Next.js 16, React 19, TypeScript, Tailwind CSS, Motion, Socket.IO Client
- **AI Models:** YOLOv8n INT8 quantized (TFLite), Hailo Voice Assistant with RAG architecture

Methodology

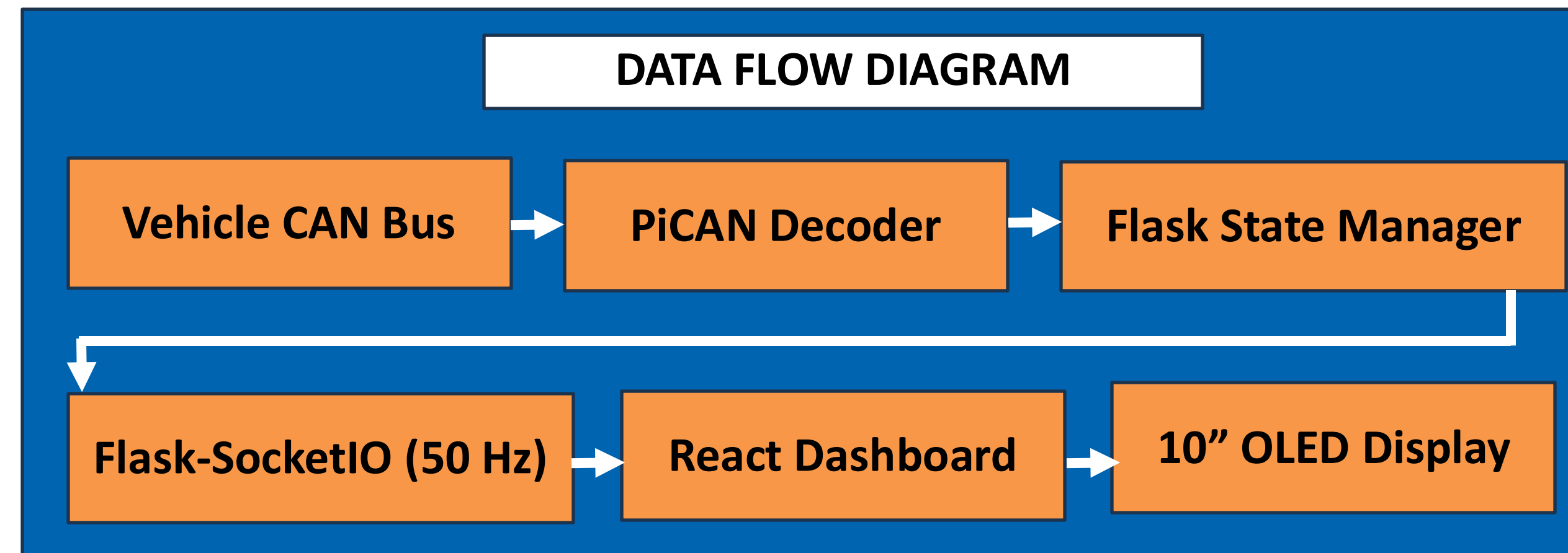


Figure 2. A PiCAN HAT (MCP2515) interfaces with the vehicle OBD-II port. The python-can library reads raw CAN frames at 50 Hz and a custom decoder parses message IDs into human-readable telemetry (speed, RPM, TPMS, gear, climate, body state). Then, a Flask + Flask-SocketIO server (gevent async) broadcasts the full vehicle state to connected clients at 30–50 Hz over WebSocket for use by all subsystems.

The system (Fig. 2) was designed using a layered architecture with four major subsystems: Hardware Layer, UI Layer, AI Layer, Navigation Layer.

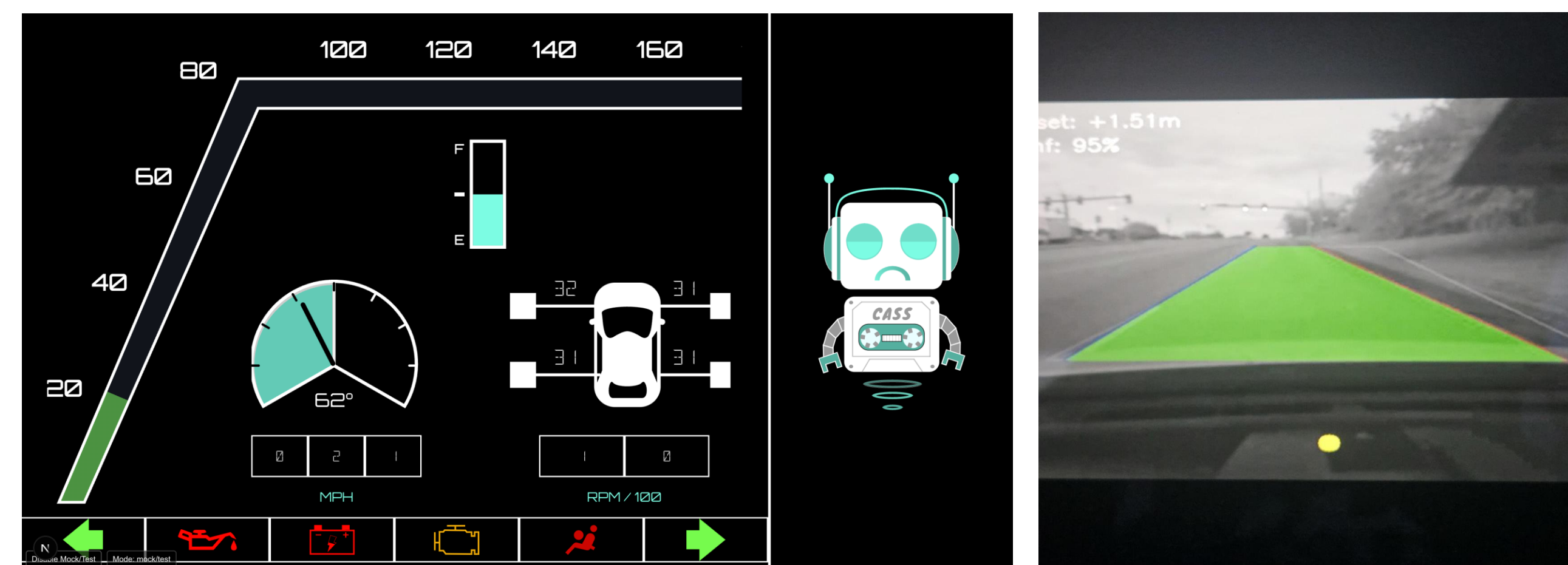
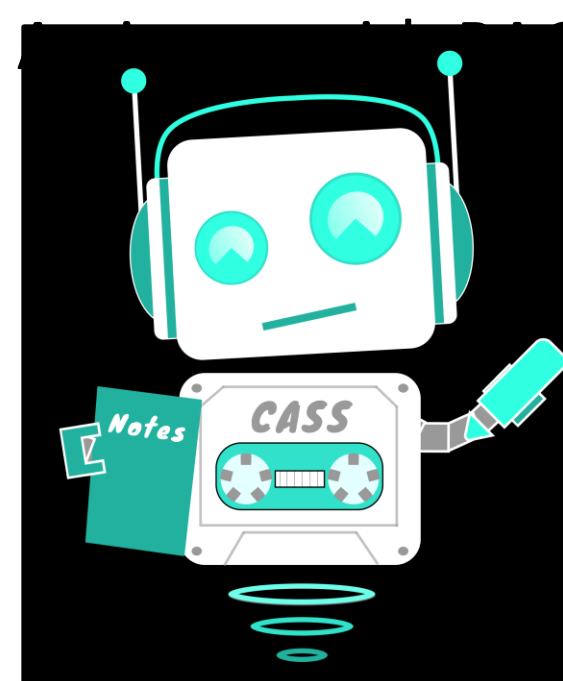


Figure 3. Cassette Drive Dashboard Interface. Warning lights flicker in red and yellow at the bottom, indicating the vehicle needs maintenance. In the middle panel, the AI assistant appears and responds to user prompts. On the far right, a vehicle navigation overlay appears to ensure safe driving.

AI Assistant

- **Hardware:** Raspberry Pi 5 (8GB), PiCAN 3 HAT, USB wide-angle camera, 10" IPS display
- **OS:** DietPi Linux, Chromium kiosk mode
- **Backend:** Python 3, Flask-SocketIO, Python-Can, OpenCV 4.8 [8], TensorFlow Lite
- **Frontend:** Next.js 16, React 19, TypeScript, Tailwind CSS, Motion, Socket.IO Client
- **AI Models:** YOLOv8n [3] INT8 quantized (TFLite), Hailo Voice architecture

Figure 4. The image on the right shows the AI assistant's "confused" expression, displayed when its response has a confused tone.



Navigation system

The Navigation Layer implements a real-time **Advanced Driver Assistance System (ADAS)** pipeline using a forward-facing USB camera processed on the Raspberry Pi 5 CPU [5] and Hailo AI HAT+ NPU.

Lane departure detection uses an 8-stage classical computer vision pipeline — grayscale conversion, CLAHE contrast enhancement, Gaussian blur, Canny edge detection [1], trapezoidal ROI masking, probabilistic Hough transform [2], slope filtering, and 15-frame temporal smoothing — achieving 94.2% accuracy at 15 FPS.

Vehicle and stop sign detection runs YOLOv8n INT8 quantized inference [3] on the Hailo AI HAT+ at 26 TOPS.

Results

- Forward collision TTC computation validated against NHTSA [6] FCW performance standards (danger threshold ≤ 2.0 s).
- All warning indicators (doors, seatbelt, check engine, TPMS, lights, turn signals) verified functional on a 2014 Toyota Camry LE.
- Full system operational on Raspberry Pi 5 with no external compute fully self-contained embedded deployment.
- AI Assistant responds to wake word, accurately transcribes user prompt, and responds through both audio and visual output with appropriate responses, drawing on live car data (speed, temperature, etc.) as necessary.

Key Performance Metrics

Metric	Target	Achieved
CAN Signals Decoded	20+	30+
CAN Update Rate	30 Hz	50 Hz
Vision Pipeline FPS	10 FPS	15 FPS
Dashboard Render Rate	30 FPS	60 FPS
Lane Detection Accuracy	90%	94.2%
Vehicle Detection Map	80%	87%
End-to-End Latency	< 500 ms	< 200 ms

Conclusions

Cassette Drive successfully demonstrates that modern digital instrument cluster capabilities including real-time telemetry, ADAS lane and collision warnings, a responsive AI assistant, and a fully animated dashboard UI can be deployed on commodity embedded hardware for under \$600 in parts.

The modular architecture cleanly separates CAN acquisition, vision processing, state management, and frontend rendering, making the system extensible to other vehicle makes and models beyond the Toyota Camry.

Lessons Learned:

- TFLite INT8 quantization is essential for real-time inference on Raspberry Pi 5
- Temporal filtering (persistence filters, moving averages) is critical to reduce false ADAS warnings
- WebSocket streaming over local LAN is sufficient for sub-200 ms dashboard latency

Future Work:

- GPS integration for navigation overlay
- Night vision with IR camera
- Coral TPU edge accelerator for improved inference speed
- Driver drowsiness detection via cabin camera
- OTA update system for remote dashboard updates

References

- [1] J. Canny, "A Computational Approach to Edge Detection," IEEE TPAMI, vol. 8, no. 6, pp. 679–698, 1986.
- [2] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines," Comm. ACM, vol. 15, no. 1, 1972.
- [3] G. Jocher et al., "Ultralytics YOLOv8," GitHub, 2023. <https://github.com/ultralytics/ultralytics>
- [4] SAE International, "OBD-II Diagnostic Standards," SAE J1979, 2014.
- [5] Raspberry Pi Foundation, "Raspberry Pi 5 Product Brief," 2023.
- [6] NHTSA, "Lane Departure Warning System Research," DOT HS 811 463, 2011.
- [7] python-can contributors, "python-can Library," <https://python-can.readthedocs.io/>
- [8] OpenCV Development Team, "OpenCV 4.x Documentation," <https://docs.opencv.org/>
- [9] Aesthetics Wiki, "Cassette Futurism," https://aesthetics.fandom.com/wiki/Cassette_Futurism