UNIVERSITÀ DI PISA

Department of Information Engineering

Master's degree in Computer Engineering

Industrial Applications

AuraBeat

Group Members:

Tiruye, Mulat Ayinet

Professors:

Pierfrancesco Foglia
Antonio Cosimo Prete

ACADEMIC YEAR 2024/2025

Abstract

This project presents **AuraBeat**, a real-time emotion recognition and mood-responsive multimedia system tailored for in-vehicle environments. The application leverages computer vision techniques to classify emotional states using live webcam input, then dynamically plays music aligned with the detected mood.

Multiple lightweight and pre-trained facial expression recognition (FER) models—including **TinyCNN** and **DeepFace**—were evaluated based on accuracy, inference latency, and suitability for real-time deployment. **TinyCNN** and **DeepFace-OpenCV** emerged as the most practical options for CPU-only systems, balancing speed and recognition quality.

AuraBeat is implemented using a modular client–server architecture built with **FastAPI** (Python) and a **ReactJS** frontend, enabling scalable, browser-based operation. Data persistence is handled via local storage, with optional logging to **MongoDB**, supporting both online and offline use.

Experimental results demonstrate the system's effectiveness in delivering timely, emotion-based feedback without the need for GPU acceleration. These characteristics position AuraBeat as a promising prototype for *affective computing* in real-world automotive scenarios.

Keywords: Facial Emotion Recognition, CNN, Real-Time Inference, Edge Computing, Affective Systems, In-Car Multimedia, Human-Centered AI

Contents

1 Introduction and Background

Emotion-aware computing systems are increasingly central to human–computer interaction, particularly in domains requiring contextual adaptation, real-time feedback, and user-centered design. Within in-vehicle environments, emotion recognition offers significant value—supporting not just safety and driver assistance, but also passenger well-being and entertainment.

AuraBeat, developed as part of an industrial applications project, addresses this opportunity by implementing a real-time, browser-based platform that detects a user's emotional state via webcam and delivers music recommendations aligned with that mood.

1.1 Emotion Recognition: Context and Significance

Emotion recognition refers to the classification of affective states from observable inputs such as facial expressions, vocal tone, or physiological signals. In vehicle interiors, facial expression stands out as a non-intrusive and practical modality for emotion sensing. Emotional states like boredom, happiness, or stress can shape passenger satisfaction and attentiveness.

Traditional automotive systems use emotion recognition to detect fatigue or distraction in drivers. However, with the shift toward autonomous and passenger-focused mobility, emotion-aware infotainment systems are becoming essential. AuraBeat fits into this new generation of intelligent media interfaces, designed to operate even in offline or constrained environments.

1.2 Emotion Modeling Approaches

AuraBeat uses both the **discrete emotion model** and the **valence–arousal model** to infer and interpret emotional states.

1.2.1 Discrete Emotion Model

Based on the work of Paul Ekman, this framework classifies emotions into fundamental categories:

- Happy
- Sad
- Angry
- Fearful
- Disgusted
- Surprised
- Neutral

AuraBeat optionally extends this set with user-friendly states such as *Excited* and *Relaxed*, which help fine-tune music recommendations. Each emotion is mapped to a corresponding genre or playlist type, enabling hands-free media personalization.

1.2.2 Valence–Arousal Model

To support finer-grained emotion classification, AuraBeat incorporates the Circumplex Model of Affect. This model represents emotions in a 2D space:

- **Valence** – degree of pleasantness (positive vs. negative)
- **Arousal** – level of intensity (calm vs. excited)

Examples include:

- High valence, high arousal → Joy, Excitement
- Low valence, low arousal → Sadness, Fatigue
- Low valence, high arousal → Anger, Anxiety

This model allows AuraBeat to detect mood trends across sessions, enabling playlist refinement and personalized feedback analysis.

1.3 Motivation for In-Car Emotion-Aware Systems

As vehicles transition toward higher automation levels, the focus shifts from driver-state monitoring to passenger experience. Static entertainment interfaces are no longer sufficient—users expect adaptive systems that respond to emotion, mood, and context.

Recent studies in automotive UX suggest that dynamic media control—based on real-time emotion—can enhance comfort, engagement, and brand differentiation. AuraBeat contributes to this emerging field by providing a browser-based, edge-friendly solution that operates without internet, accounts, or external APIs.

1.4 Project Objective and Scope

The primary objective of AuraBeat is to design and deploy an emotion-aware multimedia system capable of:

- Detecting facial emotions in real-time via webcam
- Inferring discrete emotion and valence–arousal coordinates
- Mapping each emotional state to suitable music genres
- Supporting offline operation on CPU-only environments
- Logging feedback anonymously for iterative refinement

Unlike academic systems focused purely on recognition accuracy, AuraBeat emphasizes **deployment-readiness**, **real-world usability**, and **modular design**, meeting the expectations of industrial software development.

2 System Design and Architecture

AuraBeat is designed as a modular, real-time, emotion-aware multimedia system optimized for use in constrained environments such as in-car infotainment or edge computing platforms. The architecture emphasizes separation of concerns, real-time feedback, and offline usability. The system consists of three main components—frontend, backend, and an emotion inference module—with optional database integration for persistent logging. Together, these modules enable low-latency emotion detection and adaptive music playback on CPU-only devices.

2.1 High-Level Architecture Overview

AuraBeat follows a **client–server architecture**, decoupling responsibilities between a browser-based frontend and a Python backend. The components interact via RESTful APIs to maintain modularity and support future upgrades.

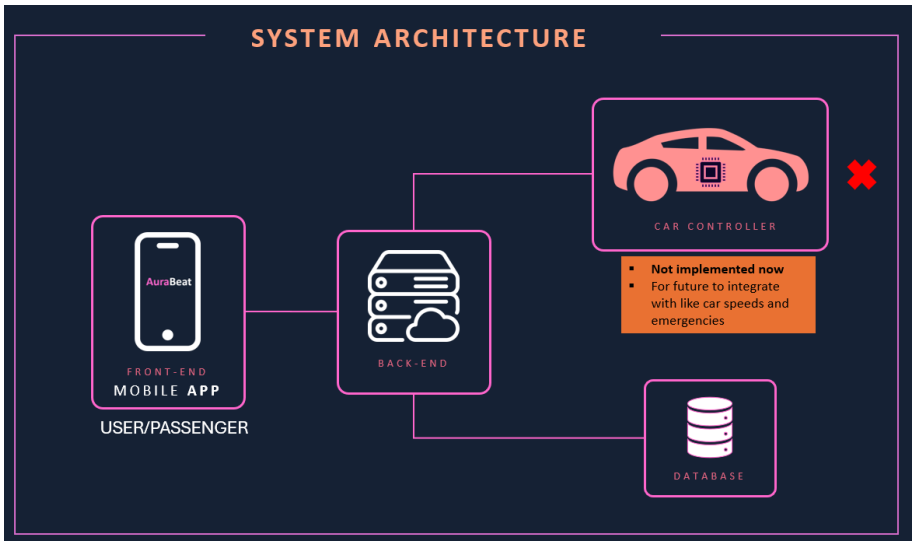

Figure 1: High-level system architecture of AuraBeat

The architecture captures webcam snapshots, runs emotion inference, maps emotions to music categories, and logs user feedback—all within a few seconds of interaction.

2.2 System Modules and Functional Components

2.2.1 Frontend (ReactJS)

Implemented using **ReactJS**, the frontend captures user input, renders results, and controls playback. Key components include:

- **Camera Component**: Uses `getUserMedia` to capture webcam frames in base64.
- **EmotionTrigger**: Initiates snapshot capture and sends it to the API.
- **Music Player**: Plays the matched track via `<audio>` element.
- **Feedback Form**: Allows users to rate the song and give optional comments.

- **Playlist Log**: Displays previously played songs and emotion logs (LocalStorage).
- **Timeline Chart**: Graphs emotion detection over time using Chart.js.
- **Admin Panel (Beta)**: Visualizes statistics like emotion frequency and feedback.

The interface is responsive, offline-capable, and designed for minimal distraction.

2.2.2 Backend (FastAPI + DeepFace)

The backend is built using **FastAPI**, which serves REST endpoints for inference and logging. It manages:

- Receiving base64 webcam images via POST `/detect-emotion`
- Running emotion classification via **DeepFace** (OpenCV backend)
- Estimating valence/arousal scores (simulated)
- Returning matched audio track information
- Logging all events to MongoDB or fallback storage

API Endpoints:

- `/detect-emotion` – emotion prediction from webcam frame
- `/log-emotion` – saves feedback and metadata
- `/vibe-count` – shows shared emotion metrics
- `/admin-stats` – exposes usage statistics

Inference latency is ∼800ms on CPU-only laptops, with support for extending the inference module in future.

2.2.3 Emotion Inference Module

The inference engine uses **DeepFace** with OpenCV backend to:

- Detect face in frame
- Classify one of 7 base emotions
- Simulate valence–arousal for finer personalization

Inputs are JPEG base64 strings, and the system returns both categorical and continuous emotional metadata.

2.3 Data Persistence and Offline Capability

AuraBeat supports hybrid storage:

- **LocalStorage**: For offline playback, cached emotion history, and logs
- **MongoDB**: Stores user feedback and session metadata when online
- **Fallback**: Defaults to "relaxed" mood in case of inference or network failure

This allows seamless operation in vehicles or intermittent network zones.

2.4 Security and Privacy Considerations

Key privacy features include:

- No raw face images are stored
- No login or user account required
- All logs are anonymous, identified by session tokens
- GDPR-aligned: minimal retention, transparent operation

Client-side inference using ONNX.js or TensorFlow.js is also considered for future updates.

2.5 Scalability and Extensibility

AuraBeat is built for extensibility:

- Drop-in support for alternate FER models
- Future valence/arousal regression via EmoNet
- Multi-user support with emotion clustering
- Personalized music logic based on user history
- Deployment on embedded boards or cloud APIs

This modular design allows AuraBeat to serve both research and commercial goals.

3 Facial Expression Recognition (FER) Models and Benchmarking

Facial expression recognition (FER) lies at the core of AuraBeat's real-time emotion-aware response system. The system was evaluated using lightweight, CPU-friendly models to ensure responsiveness and compatibility with browser-based and low-resource environments.

3.1 Model Selection Overview

Models were selected based on:

- **Classification Accuracy**: Tested on FER2013 and real webcam input
- **Inference Speed (FPS)**: Evaluated on standard CPU hardware
- **Resource Efficiency**: Minimal memory and processing footprint
- **Ease of Integration**: Seamless support within FastAPI and React-based pipeline

The following models were integrated and tested in the live prototype:

1. **TinyCNN**: A custom-trained CNN model optimized for grayscale FER, designed for fast inference and reduced size.
2. **DeepFace (RetinaFace)**: Used for face detection with emotion classification; robust but relatively slower.
3. **DeepFace (OpenCV)**: Offers faster performance on low-end CPUs, with slightly lower accuracy.

3.2 Benchmark Comparison

Model	**Accuracy (%)**	**FPS (CPU)**	**Status**
TinyCNN	54.29	11.35	Integrated
DeepFace (RetinaFace)	42.86	0.30	Integrated
DeepFace (OpenCV)	50.00	10.62	Integrated

Table 1: Evaluation of Integrated FER Models: Accuracy and Inference Speed

TinyCNN and DeepFace-OpenCV showed the best trade-off between inference speed and acceptable accuracy on CPU-only systems. RetinaFace performed reliably for detection but was too slow for interactive emotion feedback in some cases.

3.3 Emotion Classification Observations

Analysis of classification outcomes revealed:

- **High confidence for expressive states**: e.g., *Happy*, *Surprise*
- **Lower reliability for subtle emotions**: e.g., *Disgust*, *Fear*, due to fewer samples in training data

3.4 Inference Summary

All FER was performed on-device using webcam input converted to base64 JPEG. Latency remained under one second in typical use. No external model switching or GPU acceleration was used.

3.5 Next Steps (within current system limits)

- Improve recall of underperforming emotion classes via data balancing
- Explore lightweight valence–arousal regression for more nuanced control
- Tune TinyCNN further for better accuracy without size increase

AuraBeat focuses on what is practical, responsive, and reproducible—making it ready for real-world use on modest hardware without unrealistic dependencies.

4 UI and Application Integration

Beyond emotion inference, AuraBeat's strength lies in its intuitive and responsive user interface. Designed with embedded use in mind, the application offers real-time interaction through a browser-based React frontend that requires no login or persistent cloud connection. This design supports constrained environments such as in-vehicle entertainment units or offline deployments.

4.1 User Interface Design and Flow

The interface comprises five main screens, each modular and responsive:

1. Emotion Detection (Webcam Interface)
2. Music Playback View
3. Feedback Panel
4. Playlist Browser (Offline History)
5. Admin Dashboard (Developer Mode)

Emotion Detection Screen

- Captures webcam feed using `navigator.mediaDevices`.
- Snapshot is taken and encoded in base64 for backend inference.
- User triggers capture via a single "Capture Emotion" button.

4.2 Music Playback Interface

After emotion detection:

- Detected emotion, valence, and arousal are shown.
- Corresponding music track is played using HTML5 `<audio>`.
- Audio is selected from a predefined emotion-to-track mapping.

Emotion	Valence	Arousal	Track Type
Happy	High	High	Upbeat pop/electronic
Sad	Low	Low	Soft piano/acoustic
Angry	Low	High	Ambient chill beats
Relaxed	High	Low	Lo-fi instrumental
Surprised	Mid	High	Randomized genre

Table 2: Emotion–Music Mapping

4.3 Feedback Panel

Users can respond to songs via:

- Star rating (1–5)
- Optional text comment
- Like/dislike toggle

Feedback is stored:

- In `LocalStorage` when offline
- Synced to MongoDB when online
- Logged with emotion, song, time, and session token

4.4 Offline Playlist and History

AuraBeat includes offline storage using `LocalStorage`. Users can:

- Replay previously played songs
- View emotion–song history
- Delete or re-rate songs

Stored metadata includes emotion, timestamp, song ID, and prior rating.

4.5 Admin Dashboard

A minimal dashboard at `/admin` supports developer inspection:

- Displays emotion frequency distribution
- Graphs valence/arousal over time
- Lists feedback with timestamps
- Shows model-specific FPS metrics

Charts are built with Chart.js and auto-refresh periodically.

4.6 Frontend–Backend Synchronization

Frontend sends data to Flask backend via REST:

```
POST /detect-emotion
Payload: { image: "<base64-string>" }

← Response:
{ emotion: "happy", valence: 0.82, arousal: 0.71, song: "happy.mp3" }
```

Other endpoints include:

- `POST /log-emotion` — logs current detection
- `GET /session-log` — fetch emotion timeline
- `GET /vibe-count` — retrieve crowd emotion stats

4.7 User Experience and Accessibility Goals

AuraBeat was designed with the following priorities:

- Sub-2s emotion-to-music latency
- One-click interaction for core tasks
- Mobile-first responsive layout
- Dark theme for low-light vehicle usage
- No login required (anonymous session)

Keyboard shortcuts, enlarged UI targets, and future ARIA tags are under evaluation for WCAG compliance.

5 Prototype Design and Deployment

The AuraBeat prototype was developed as a browser-based, real-time emotion-to-music application designed for general-purpose machines without GPU acceleration. It runs entirely within a local or network-connected client–server architecture, making it lightweight and suitable for environments with limited resources such as vehicles or mobile infotainment systems.

5.1 Technology Stack and Environment

The prototype system was tested on a standard laptop (Intel i5, 8GB RAM, no GPU). The software stack includes:

- **Frontend**: ReactJS (JavaScript), HTML5 APIs, Chart.js
- **Backend**: Flask (Python), DeepFace API, NumPy, Sklearn

- **Database**: MongoDB (optional); browser `LocalStorage` as fallback
- **Hardware**: Integrated webcam; no dedicated accelerator

The full system is platform-independent and runs inside a modern browser with webcam support. While edge deployment (e.g., Raspberry Pi) is not yet implemented, the application is structured to support such integration in the future.

5.2 Runtime Execution Flow

1. User accesses the web interface; webcam permission is requested.
2. A snapshot is captured via `getUserMedia()` and converted to base64.
3. The frontend sends the image to the backend via a POST request to `/detect-emotion`.
4. The backend infers the emotion using TinyCNN or DeepFace (OpenCV backend).
5. Detected emotion, valence, and arousal are returned as JSON.
6. Frontend maps the result to a song and plays it using the built-in audio component.
7. Feedback (rating, comment) is optionally submitted to local or cloud storage.

5.3 Performance Metrics

Testing was conducted under real-use conditions, using only CPU resources. Results are summarized in Table 3.

Stage	**Avg Time (ms)**	**Notes**
Webcam Capture	120	Base64 conversion included
TinyCNN Inference	850	Fully CPU-based
DeepFace (OpenCV) Inference	1350	Slower but usable
Music Mapping + UI Rendering	250	Audio load + DOM updates
Total Latency (TinyCNN)	**∼1.2s**	End-to-end delay

Table 3: Prototype Performance Breakdown

The system maintained real-time responsiveness (<1.5s) on average, even without hardware acceleration.

6 Conclusion and Future Directions

AuraBeat demonstrates the feasibility of real-time, emotion-aware multimedia systems designed for constrained environments such as in-car or edge-device deployments. Despite hardware limitations and lack of GPU acceleration, the prototype successfully integrated facial emotion recognition, automatic music response, and user feedback into a cohesive loop—entirely within a browser-based setup.

The system's modularity and simplicity were key strengths: users experienced minimal latency, intuitive interactions, and full offline capability. Though some components (e.g., valence/arousal prediction and multi-user handling) were simulated or simplified, the overall experience remained engaging and functionally robust.

Future improvements include:

- Deploying advanced models (e.g., RMN, ViT) on GPU-enabled or cloud platforms
- Integrating true valence–arousal predictors (e.g., EmoNet or custom regressors)
- Supporting multi-user detection and emotion fusion
- Enhancing personalization via optional login and adaptive recommendations
- Migrating inference to client-side using ONNX.js or similar frameworks
- Adding privacy controls such as data retention toggles and visible consent cues

With these upgrades, AuraBeat can evolve into a deployable affective computing toolkit for automotive, wellness, and educational domains.