VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY

FINAL REPORT

COMPUTATIONAL THINKING- CS117.P21

REAL-TIME DRIVER STATE RECOGNITION VIA HEALTH INDICATORS AND FACIAL SIGNALS

Lecturer : PhD. Ngo Duc Thanh

Author : Le Ngoc Thanh – 23521443
Truong Hoang Thanh An - 23520032
Nguyen Xuan An – 23520023
Vu Viet Hoang – 23520548
Mai Thai Binh – 23520158

Ho Chi Minh City, June 13, 2025

SUPERVISOR'S EVALUATION AND FEEDBACK

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

HCM city, July 13, 2025

Lecturer

PhD. Ngo Duc Thanh

ACKNOWLEDGEMENTS

First and foremost, I, Truong Hoang Thanh An – group leader of Computational Thinking Group 14 – would like to express our deepest and most sincere gratitude to Dr. Ngo Duc Thanh, our academic advisor, for his dedicated guidance and support throughout the course of this project. His invaluable direction and insightful technical feedback not only enabled our team to structure and complete the project in a scientific and logical manner, but also helped each member to develop critical thinking, sharpen research skills, and gain a deeper understanding of core computer science principles—particularly within the context of our chosen topic: **Real-Time Driver State Recognition via Health Indicators and Facial Signals**.

We would also like to thank our classmates for their encouragement, constructive suggestions, and resource sharing, which greatly supported our group during the development of this project. Special thanks go to all team members for their close collaboration, professionalism, and strong sense of responsibility that allowed us to complete the work on time and to the best of our ability.

Despite our utmost effort, due to the limited timeframe and our still-developing expertise, we acknowledge that this report may contain shortcomings. We sincerely welcome any feedback or suggestions from instructors and peers to help us further improve and expand the project in future iterations.

Once again, we truly appreciate Dr. Thanh's mentorship and unwavering support—his guidance was a key factor in the successful completion of this project.

CONTENTS

CHAPTER 1.
PROBLEM IDENTIFICATION

1.1 Problem Clarification

Driver drowsiness and inattention have emerged as critical factors contributing to traffic accidents, especially with the growing density of urban traffic and prolonged commuting hours. Numerous studies have consistently shown that fatigue-related impairment is among the leading causes of severe road incidents. A particularly concerning aspect is that drivers often remain unaware of their own drowsiness until it results in a critical safety failure.

While existing commercial driver monitoring systems offer some level of support, they are often limited in scope. Most systems rely solely on visual cues such as eyelid closure or head posture, or exclusively on physiological indicators like heart rate or skin conductance. These unimodal approaches frequently lack the robustness required to generalize across diverse driving conditions and driver profiles, leading to inconsistent performance in real-world applications.

To address these limitations, we propose the development of a real-time driver alertness recognition system that combines both external visual indicators and internal physiological signals. By integrating facial video analysis with biometric data, the system aims to provide a more holistic, accurate, and reliable assessment of a driver's alertness state.

1.2 Problem Definition

The problem is formally defined as follows:

We aim to design and implement a real-time binary classification system capable of determining whether a driver is in an alert or drowsy state, based on two synchronized data streams: facial video and heart rate signals.

To meet this objective, the system must fulfill the following criteria:

- It must be able to continuously capture and process facial video footage from a front-facing camera mounted on the dashboard.
- Simultaneously, it should receive real-time heart rate (HR) measurements from a wearable biometric sensor.
- For each video frame, the system should output a corresponding binary label, where the value **1** indicates the driver is alert, and **0** indicates the driver is drowsy.
- Each frame must be processed with a maximum latency of less than one second.
- The system should maintain an average classification error of less than one percent under standard testing conditions.

1.3 Objectives

The primary objectives of this project include:

- Developing a deep learning model that integrates multimodal input sources—specifically, facial imagery and heart rate data—to accurately classify driver alertness in real time.
- Ensuring the system can efficiently process synchronized input streams with high throughput and stability.

- Meeting strict real-time performance requirements, including sub-second processing latency and high classification accuracy.
- Optimizing the architecture for deployment on lightweight, resource-constrained environments such as embedded systems.
- Embedding data privacy safeguards throughout the system, especially in how biometric information is processed and stored.

1.4 Input and Output Specification

Input specifications:

- A real-time RGB video stream of the driver's face, captured using a frontal-facing dashboard camera.
- Synchronized heart rate data (expressed in beats per minute) acquired from a wearable sensor, aligned to the timestamps of the video frames.

Output specifications:

- A sequence of binary classification labels that correspond to each frame of the input video. Specifically:
    - A label of **1** indicates that the driver is in an alert state.
    - A label of **0** indicates that the driver is drowsy.

1.5 Functional Requirements

The system is expected to fulfill the following functional requirements:

- It must be capable of continuously collecting and processing facial video data and heart rate signals in real time.
- The classification output for each frame must be generated within a processing time not exceeding one second.
- The system's architecture should support easy integration with downstream alert systems, such as in-vehicle auditory or visual warnings.
- The model should be designed to allow for retraining with expanded datasets, without requiring structural changes to the overall architecture.

1.6 Constraints

The system must operate under the following constraints:

- **Latency constraint:** The maximum processing time for each frame must be less than one second.
- **Accuracy constraint:** The classification error must remain below one percent under standard operating conditions.
- **Environmental scope limitations:** The system does not explicitly account for challenging external conditions such as low-light environments, motion blur, facial occlusions, or reflective surfaces (e.g., glasses).
- **Data locality requirement:** All data should be processed locally on the device to ensure user privacy. No data is to be transmitted externally unless explicitly authorized by the deployment scenario.

1.7 Scope

In-scope functionality:

- Real-time processing and classification based on synchronized facial video and heart rate input.
- Binary classification of the driver's alertness state for each frame.
- Offline training and real-time inference within the constraints of standard consumer hardware.
- Evaluation of the model using controlled or simulated driving scenarios.

Out-of-scope considerations:

- The use of infrared imaging or additional biometric signals such as EEG or electrodermal activity.
- Modeling of environmental influences such as weather conditions, road surface, or ambient noise.
- Full system-level integration with vehicle control systems or third-party hardware platforms.

CHAPTER 2.
THE PROCESS OF APPLYING COMPUTATIONAL THINKING

2.1 Introduction

In this project, Computational Thinking is not merely treated as a programming technique, but as a principled design methodology that enables us to address a real-world, safety-critical problem. Computational Thinking provides the cognitive scaffolding for structuring a solution that balances algorithmic accuracy with real-time constraints, interpretability, and operational simplicity.

We aim to develop a real-time system capable of detecting driver drowsiness using only two input modalities: heart rate (HR) signals and frontal facial video. The system must operate with high temporal fidelity—outputting a stream of binary predictions (1 = alert, 0 = drowsy) aligned to incoming video frames—while maintaining strict performance thresholds: under 1 second of processing time per frame and a prediction error below 1%.

Using Computational Thinking, we decomposed the problem into discrete functional units, identified behavioral and physiological patterns indicative of drowsiness, and abstracted away irrelevant environmental complexities. This allowed us to design an efficient and modular architecture tailored to high-speed, real-world deployment.

2.2 graphic organizer

2.2.1 Iteration 1 – Identifying the Real-World Problem

Problem Identification

Driver fatigue remains a major contributor to traffic accidents globally, not due to mechanical failures or road conditions, but often due to drivers falling asleep unintentionally. Alarmingly, fatigue is frequently unrecognized until it's too late. This observation motivated our central research question:

Can we build a real-time system capable of reliably detecting driver drowsiness using only minimally invasive input streams?

We hypothesized that a combination of facial expression cues and HR signals could offer sufficient evidence for real-time classification of driver alertness.

Decomposition

At the initial design stage, we realized that the problem was non-trivial and required functional decomposition to make development tractable:

- First, the facial video stream must be parsed frame-by-frame to detect external signs of fatigue such as yawning, eye closure, and reduced facial motion.
- Separately, heart rate variability provides internal physiological indicators of drowsiness, such as a drop in average BPM and reduced fluctuations.

Both modalities are processed independently: facial frames are passed through a deep vision model to extract visual embeddings, while HR data is normalized and encoded via a simple neural network. The resulting feature vectors are fused and evaluated jointly to make a final binary classification decision.

Pattern Recognition
Analysis of annotated datasets revealed recurrent patterns in both modalities. Drowsy states were marked by consistent external behaviors and subtle but statistically significant HR changes. This cross-modal consistency enabled our model to learn discriminative features that generalized well across individuals.

Abstraction
We formulated a minimalistic decision space, reducing the output to a binary state: non-drowsy or drowsy. Complexities such as environmental lighting, clothing, or facial accessories were intentionally abstracted away. This focused approach ensured robustness and reduced model overhead without sacrificing accuracy.

2.2.2 Iteration 2 – Modeling the Pipeline

Problem Refinement
Having established a general design, we moved on to the technical challenge of synchronizing the two real-time input streams—video and HR data—while keeping latency sufficiently low for real-time applications. Furthermore, we wanted the system to be model-agnostic to users, avoiding dependence on individualized training.

Decomposition
The processing pipeline was structured into three stages:

1. Image Processing – Each frame from the facial video stream is preprocessed and passed into a Vision Transformer to extract high-level visual embeddings.
2. Heart Rate Encoding – HR values corresponding to each frame timestamp are processed through an MLP, generating compact physiological feature vectors.
3. Multimodal Fusion and Classification – The two feature sets are concatenated and passed into a lightweight classifier that outputs a binary prediction for each frame.

The output is a continuous stream of 0 or 1 labels that align temporally with the incoming video frames.

Pattern Recognition
Through experimentation, we noticed that specific facial cues (e.g., slight eye openings, head tilts) frequently coincide with a decrease in HR. This multimodal correlation greatly improves model reliability compared to using any single source. By learning such spatiotemporal patterns across modalities, the model can infer states more robustly.

Abstraction
Rather than processing long temporal sequences, we opted to process each video frame and corresponding HR value independently. This reduces computational overhead while

maintaining responsiveness. Moreover, all non-essential attributes—like driver clothing, background clutter, or lighting shifts—were excluded from the model. Only physiologically and behaviorally meaningful features are retained for inference.

2.2.3 Iteration 3 – Final Task Formulation and Constraints

Problem Identification

With a working prototype in place, we turned our focus to operational constraints and system requirements. Given the real-time nature of the task, **inference speed** and **accuracy** became the most critical concerns. A slow system risks delivering alerts too late to be actionable, while a low-precision model may lead to false positives, eroding trust and usability.
Thus, we imposed two hard requirements:

- Processing time per frame must be under 1 second.
- Prediction error must remain below 1% under normal conditions.

Decomposition

To meet these requirements, we optimized every stage of the pipeline.

- Facial images were resized and passed through a lightweight ViT architecture with reduced depth to accelerate inference.
- HR data was kept minimal—only raw heart rate was used, avoiding extra physiological parameters to reduce computation.
- Feature fusion and classification were streamlined via a shallow MLP instead of using more resource-intensive mechanisms like attention or recurrent units.
- This lean architecture ensured both speed and simplicity while preserving classification performance.

Pattern Recognition

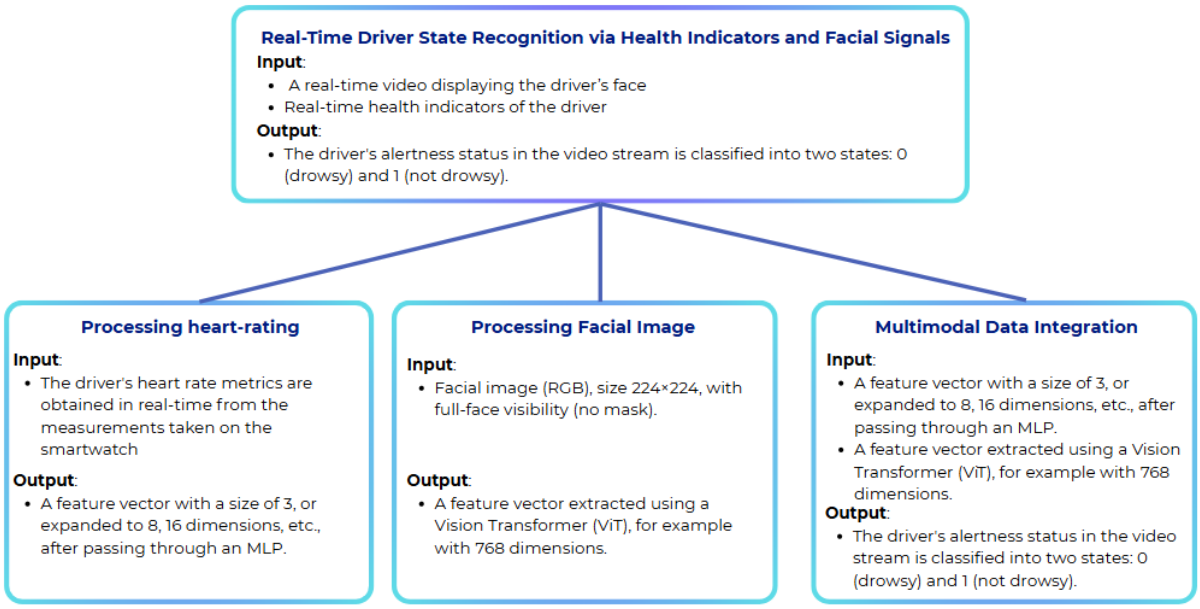
Model training revealed that even minimal features, when properly aligned, could yield highly reliable results. Eye behavior and HR dips jointly served as strong drowsiness indicators. By focusing on these joint signals, the system maintained accuracy without increasing model complexity.

Abstraction

The entire system was formalized as a real-time binary function: (face image, HR) → {0,1}. Historical context was not retained; each decision was made per-frame, with no temporal memory or cloud processing. This abstraction guarantees both privacy and responsiveness, aligning with real-world operational needs.

2.3 Decomposition tree

The decomposition tree below outlines the hierarchical architecture of our system. It visually divides the solution into three autonomous modules: (1) processing facial images, (2) analyzing heart rate signals, and (3) integrating features for classification. This modularity promotes parallel processing, fast inference, and system scalability.



2.4 Computational Thinking Rationale

The core objective of this project is to design and implement a system that can detect driver drowsiness in real-time using only two synchronized input streams: frontal facial video and heart rate signals. The system must produce an output stream of binary predictions (0 for drowsy, 1 for alert) aligned with each video frame, all while satisfying strict technical constraints—namely, a processing time of under 1 second per frame and an accuracy error below 1%.

To achieve this, we grounded our development process in the framework of Computational Thinking. Rather than tackling the problem monolithically, we adopted an iterative approach that decomposed the system into manageable components, recognized recurring behavioral and physiological patterns, and abstracted away unnecessary complexity.

2.4.1 Decomposition

Decomposition was critical in our early design. By dividing the system into three primary functional modules—facial image processing, heart rate signal processing, and multimodal classification—we could isolate and independently optimize each part. This modular architecture also made the system easier to debug, test, and improve.

2.4.2 Pattern recognition

Pattern recognition enabled the model to detect subtle but consistent correlations between visual indicators of fatigue (e.g., drooping eyelids, yawning) and physiological patterns (e.g., reduced heart rate variability). By training on labeled data, the system learned to associate these patterns with drowsy or alert states, significantly boosting prediction accuracy.

2.4.3 Abstraction

Abstraction allowed us to strip away irrelevant environmental factors—such as lighting variation or video blur—that could distract or overload the model. By focusing only on the core, meaningful inputs (heart rates and face), we reduced computational load and maintained high-speed performance, which is critical for real-time use cases.

2.5 Summary of the Computational Thinking Process

Building a real-time driver alertness detection system required more than just technical tools—it demanded a structured thought process. We relied on computational thinking as a practical framework to define, decompose, and implement each part of the solution effectively.

Step 1 – Abstracting the Core Problem

We started by reframing the real-world issue. Instead of trying to capture every possible driver behavior, we simplified the goal to a binary task: determine whether the driver is "alert" or "drowsy."

To keep the system lightweight and deployable, we focused on just two input sources: video of the driver's face and their heart rate. This helped narrow the scope and made the problem computationally manageable.

Step 2 – High-Level Decomposition

From there, we broke the task into three essential parts:

1. Process facial images to detect signs of fatigue.
2. Analyze heart rate signals as a measure of internal state.
3. Combine both to decide the driver's level of alertness.

This breakdown allowed us to focus on one part at a time, making design and testing easier and more efficient.

Step 3 – Refining Each Subsystem

We then detailed the steps within each part. For the visual pipeline: detect the face, resize it, split into patches, and extract features using a Vision Transformer.

For heart rate: normalize the signal, pass it through a simple MLP, and generate a compact vector.

Finally, we fused both outputs into one and fed them into a classifier that gives us a prediction per video frame.

Step 4 – Pattern Recognition

By analyzing our data, we noticed repeating signs: drowsy drivers tend to blink slowly, show less facial motion, and have more stable, sometimes lower heart rates.

These patterns were used to train the model, making it better at distinguishing between alert and drowsy states.

Step 5 – Mapping to Known Solutions

We matched each part of the system with reliable tools.

The Vision Transformer handled facial data; a shallow MLP processed the HR signals; and a simple fusion network combined them for classification.

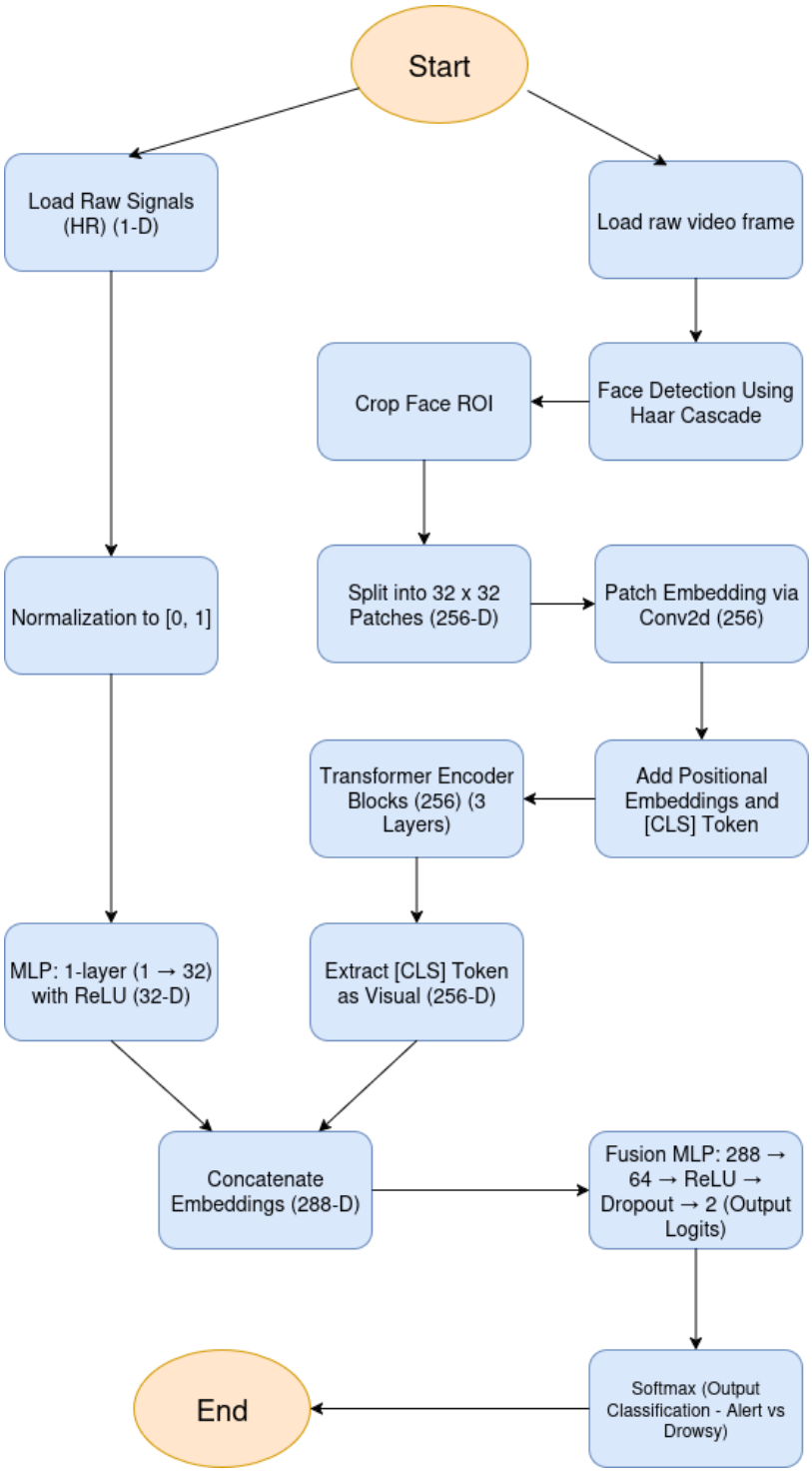
This use of well-understood techniques helped us build a dependable system faster and with fewer risks.

Step 6 – Simplifying Under Constraints
With hard limits on processing time (under 1 second per frame) and accuracy (less than 1% error), we refined the system down to essentials. Irrelevant variables—such as lighting variance, facial accessories, or environmental noise—were intentionally excluded from training. By focusing on a minimal yet effective feature set, we preserved speed without sacrificing precision, ensuring the system is viable for real-time use even on modest hardware.

Overall, this CT-driven process allowed us to move from a complex real-world issue to a clear, effective solution—step by step, with logic and clarity at every stage.

CHAPTER 3.
SOLUTION

3.1 Algorithm Flowchart



3.2 Detailed Steps

3.2.1 Start

- Begin the process of analyzing video and biometric data to detect drowsiness.
- This is the starting point where the system initializes components such as the camera and the deep learning model. The process begins by opening the video stream and loading the classification model.

3.2.2 Sub-problem 1: Processing Facial Image

Load Raw Video Frame

- Load a raw frame from the input video, typically from a webcam.
- Frames are continuously captured from the webcam for real-time processing, ensuring the input data is always current

Face Detection Using Haar Cascade

- Use the Haar Cascade method to detect a face within the frame.

Formula:

$$\text{decision} = \sum_{i=1}^{N} \alpha_i h_i(x) > \theta$$

Where:

- $h_i(x)$: A weak classifier based on Haar features.
- α_i: The weight of the weak classifier.
- θ: The decision threshold.

- Haar Cascade uses rectangular features and AdaBoost to identify the facial region, which helps focus processing on the most important area

Crop Face ROI

- Crop the Region of Interest (ROI) containing the face from the frame.
- The facial area is isolated to reduce noise from the background and increase processing efficiency.

Split into 32x32 Patches (256-D)

- Divide the face image into 32x32 pixel patches

$$\text{patch\_embed} = \text{Conv2d}(x, \text{in\_channels} = 3, \text{out\_channels} = 256, \text{kernel\_size} = 32, \text{stride} = 32)$$

$$x = \text{patch\_embed.flatten}(2).\text{transpose}(1,\ 2)$$

- The image is divided into 49 patches (for a 224x224 image), each embedded into a 256-dimensional vector for processing by the Transformer.

Patch Embedding via Conv2d (256)

- Embed the patches using a Conv2d layer, resulting in a 256-dimensional embedding.

- This is part of the patch splitting process, using Conv2d to convert image patches into a numerical representation.

Add Positional Embeddings and [CLS] Token

- Add a [CLS] token and positional embeddings to the patch embedding vectors.

$$\text{class\_token} = \text{self.class\_embedding.expand(batch\_size, -1, -1)}$$

$$x = \text{torch.cat}((\text{class\_token}, x), \text{dim} = 1)$$

$$x+ = \text{self.position\_embedding}$$

- The [CLS] token serves as an aggregate representation of the image, while positional embeddings help the model understand the sequence of the patches.

Transformer Encoder Blocks (256) (3 Layers)

- Apply 3 layers of Transformer Encoders to process the embedded vectors.

  - Multi-Head Self-Attention (MSA):

$$\text{MSA}(x) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

$$\text{head}_i = \text{softmax}\left(\frac{(xW_i^Q)(xW_i^K)^T}{\sqrt{d_k}}\right)(xW_i^V)$$

  - MLP:

$$\text{MLP}(x) = \text{GELU}(xW_1 + b_1)W_2 + b_2$$

  - Residual connections:

$$x = x + \text{MSA}(\text{LayerNorm}(x))$$

$$x = x + \text{MLP}(\text{LayerNorm}(x))$$

- The Transformer Encoder processes the relationships between patches and the [CLS] token, creating a powerful composite representation.

Extract [CLS] Token as Visual (256-D)

- Extract the [CLS] token as the 256-dimensional visual representation.
- The [CLS] token is used as the final output of the Vision Transformer, representing the entire image.

$$\text{visual\_embed} = x[:, 0]$$

3.2.3 Sub-problem 2: Processing heart-rating

Load Raw Signal (HR) (1-D)

- Load raw biometric data: Heart Rate (HR).
- HR data provides supplementary physiological information to increase classification accuracy.

Normalization to [0, 1]

- Normalize the HR signal to the range [0, 1]

$$\text{norm\_hr} = \frac{\text{hr} - 60}{100 - 60}$$

- Normalization ensures the HR signal has a scale from 0 to 1, helping the model to process it uniformly.

MLP: 1-layer (3 → 32) with ReLU (32-D)

- Apply a 1-layer MLP with a ReLU activation function to the HR signal.

$$\text{physio\_embed} = \text{ReLU}(W \cdot \text{norm\_hr} + b)$$

Where W (1x32) and b (32) are the weights and biases.

- The MLP transforms the HR data into a 32-dimensional representation, adding non-linearity through ReLU.

3.2.4 Sub-problem 3: Multimodal Data Integration

Concatenate Embeddings (288-D)

- Concatenate the visual (256-D) and biometric (32-D) embeddings into a single 288-dimensional vector.

$$\text{fused\_embed} = \text{concat}(\text{visual\_embed}, \text{physio\_embed})$$

- Concatenation combines information from the two data sources to create a unified representation.

Fusion MLP: 288 → 64 → ReLU → Dropout → 2

- Apply a fusion MLP to generate logits for classification.

- Layer 1:
$$h = \text{ReLU}(W_1 \cdot \text{fused\_embed} + b_1)$$
- Dropout:
$$h = \text{Dropout}(h, p = 0.3)$$
- Layer 2:
$$\text{logits} = W_2 \cdot h + b_2$$

- The fusion MLP reduces the data dimensionality and produces output for the two classification classes. Dropout is used to reduce overfitting.

Softmax (Output Classification - Alert vs Drowsy)

- Apply the Softmax function to classify the state as ”Non-drowsy” or ”Drowsy”.

$$p_i = \frac{e^{\text{logits}_i}}{\sum_{j=1}^{2} e^{\text{logits}_j}}, \quad i = 1, 2$$

$$\text{pred\_class} = \arg\max_i p_i$$

- Softmax converts the logits into probabilities. The class with the highest probability is chosen as the prediction.

End

- Conclude the processing for the current frame.
- The process repeats for each subsequent frame until the system is stopped, at which point resources are released.

CHAPTER 4.
DATASET AND EVALUTION

4.1 Datasets

To enable multimodal real-time alertness classification, we combined two complementary public datasets—one for facial image data and another for physiological signals.

a. Driver Drowsiness Detection Dataset

This dataset contains **9,120 RGB facial images** captured from frontal-facing cameras, featuring drivers seated upright without any facial occlusion (e.g., no masks or sunglasses). The dataset is manually balanced and split into two distinct categories:

- **Active**: 4,560 images of alert drivers.
- **Fatigue**: 4,560 images of drowsy or inattentive drivers, exhibiting signs such as closed eyes, yawning, or facial slouching.

To standardize indexing and support synchronized integration with biometric data, all image files were renamed using the format Active_{idx}.jpg or Fatigue_{idx}.jpg, with sequential indices starting from 0.

b. WESAD Dataset

The WESAD (Wearable Stress and Affect Detection) dataset provides rich multi-sensor physiological recordings from **15 participants**, collected via two wearable devices: RespiBAN (chest-worn) and Empatica E4 (wrist-worn). While WESAD includes a wide range of biosignals (e.g., EDA, TEMP, BVP, ECG), we retained only **Heart Rate (HR)** values to align with real-world wearable constraints and our system's design scope.

To match the facial dataset, we selected **4,560 HR samples within the normal range (60 ≤ HR ≤ 100)** for the Active class and **4,560 extreme HR samples (HR < 60 or HR > 100)** for the Fatigue class. These were compiled into a unified HR.csv file containing 9,120 entries, ordered to correspond exactly with the renamed image dataset.

c. Dataset Summary and Justification

This integrated dataset captures both **external behavioral cues** (through facial expressions) and **internal physiological signals** (via heart rate), offering a comprehensive foundation for robust drowsiness detection in real-time applications. The careful balancing, normalization, and one-to-one synchronization of both data streams ensures high data integrity and precise frame-level alignment between modalities.

Although the WESAD dataset provides a wide spectrum of biosignals (e.g., EDA, TEMP, BVP, RESP), only **heart rate (HR)** was selected for this project due to its simplicity, stability, and widespread support across commercial wearable devices such as smartwatches. This decision

reduces system complexity, enhances real-time feasibility, and aligns with practical deployment constraints.
To ensure both **class balance** and **temporal consistency**, we constructed a custom HR.csv file with exactly **9,120 entries**. The first 4,560 values, corresponding to the **Active class**, represent typical resting HR values (60–100 BPM). The remaining 4,560 values reflect **Fatigue conditions**, defined by out-of-range HR readings (<60 or >100 BPM). These thresholds were informed by basic medical standards, facilitating clean and interpretable class boundaries.
In sum, this dual-source dataset forms a well-aligned, multimodal training set that enables the model to learn both **surface-level visual fatigue markers** and **underlying physiological rhythms**, significantly enhancing performance under diverse real-world conditions.

4.2 Evaluation

The system is evaluated based on the following criteria:

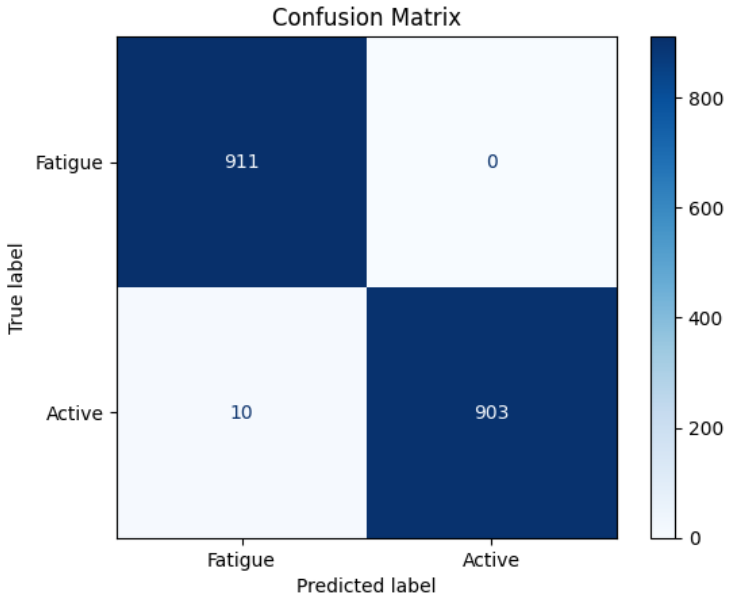
Requirements	Evaluation Metrics
Accurate recognition of driver state	Accuracy, F1-score
Minimization of false or missed alerts	Precision, Recall

After determining the two main requirements for our task (processing time under 1 second and accuracy above 99%), we established the following metrics:

- Accuracy, F1-score: Target below 1% error, used to evaluate the accuracy of driver state recognition.
- Precision, Recall: Used to minimize false or missed alerts.

To calculate metrics such as F1-score, Precision, and Recall, the dataset was split into train_dataset and test_dataset with an 8:2 ratio.

After training and testing, we used a Confusion Matrix to provide a detailed view of the model's performance by showing the number of correct and incorrect predictions for each class. For a binary classification problem (2 classes), the confusion matrix is a 2x2 matrix, where diagonal elements represent correct predictions, and off-diagonal elements indicate misclassification errors.
Implementation typically involves using the confusion_matrix function from scikit-learn to compute the matrix, with results visualized as a heatmap using seaborn/matplotlib.
Results after training and testing:



From the confusion matrix, we derived the following Classification Report:

Class	Precision	Recall	F1-score	Support
Fatigue	100%	100%	100%	911
Active	99%	99%	99%	913
Accuracy			99%	1824

The Classification Report provides detailed metrics for each class, including precision, recall, and F1-score, calculated as follows:

- Precision: The ratio of correct predictions among samples predicted as belonging to a class:

  Precision = True PositivesTrue Positives + False Positives
- Recall: The ratio of samples of a class correctly predicted:

  Recall = True PositivesTrue Positives + False Negatives

  F1-score: The harmonic mean of precision and recall:

  F1-score = 2 x Precision x RecallPrecision + Recall

Observations:

- Accuracy = TP + TNTP + TN + FP + FN = 911 + 903911 + 903 + 0 + 10 = 0.9945. This indicates that the model achieves very high accuracy, suitable for the task.
- No misclassifications for the Fatigue class, meaning all negative cases (unfit to drive) were predicted correctly, increasing real-world reliability.

- The error rate for the Active class is very low, with only 10 misclassified labels. Misclassifications between Active and Fatigue are not dangerous and are acceptable.

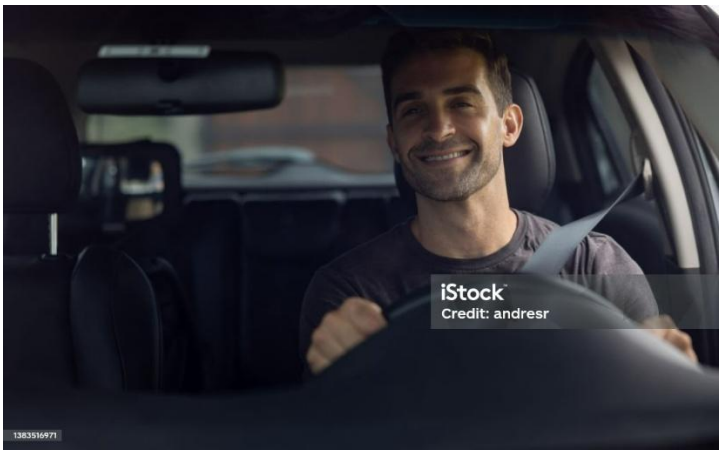
4.3 Performance Results & Analysis

4.3.1. Performance Results

a. Manual Testing on Samples

Results from test_model_on_samples with 3 samples (all are images):

Sample 1: Predicted 1 (Active), confidence 73.27%, true label 1.



Sample 2: Predicted 1 (Active), confidence 79.33%, true label 1.

Sample 3: Predicted 0 (Fatigue), confidence 83.66%, true label 0.

Prediction confidences range from 73–85%, indicating varying reliability depending on the sample.

b. Real-Time Performance

In video application (test/1.mp4): Frame-by-frame processing with face detection and state prediction demonstrates near real-time capability.

4.3.2. Analysis

a. Overall Performance

Strengths:

- High accuracy (99.45%): The model performs exceptionally well on the test dataset, critical for driver monitoring applications.
- Perfect Fatigue detection: 100% Precision and Recall ensure no Fatigue cases are misclassified as Active, maximizing safety.
- Balanced F1-score: Near 100% for both classes indicates a good balance between Precision and Recall.

Limitations:

- Errors in Active class: 10 cases misclassified as Fatigue (False Positives), ~1.095% of 913 Active samples. While low and not dangerous, this could lead to false alerts in practice.

b. Generalization

- The current model does not apply augmentation (e.g., RandomHorizontalFlip, RandomRotation), potentially reducing generalization to new data, especially if real-world data includes different angles or variations.
- Manual testing on 3 samples shows prediction confidences varying (73%–86%), reflecting dependence on image and physiological data quality.

c. Real-World Application

- Face Detection: Using haarcascade_frontalface_default.xml is suitable for real-world environments but may struggle with complex angles or low lighting.

- Real-Time Performance: Displaying FPS indicates potential for continuous monitoring, but further optimization is needed to achieve higher FPS (e.g., >30FPS) for a smoother experience.
- Static Physiological Data: Fixed EDA, HR, and Temp values in videos do not reflect real-time changes, potentially reducing accuracy in dynamic scenarios.

4.3.3. Conclusion

The current model demonstrates impressive performance with an accuracy of ~99.45%, particularly strong in detecting the Fatigue class (100% Precision and Recall). However, discrepancies in reported metrics and minor errors in the Active class warrant further investigation. With video processing capabilities and real-time potential, the model is suitable for driver monitoring applications but requires improvements to enhance generalization and real-world performance.

4.4 Real-Time Testing (Video, FPS)

4.4.1. Configuration and Setup

Model and Device: Uses CustomViTFusionModel with 2 output classes ("drowsy" and "non drowsy"), running on GPU (cuda) if available, optimizing real-time processing performance.
Video Source: Uses live camera input via cv2.VideoCapture(0), enabling real-time video stream processing from a webcam instead of a video file as in demo.py.
Image Preprocessing:

- Applies transforms.Compose with Resize((224, 224)) and ToTensor() to standardize image size and format, suitable for the Vision Transformer model.
- Face detection is performed using cv2.CascadeClassifier with haarcascade_frontalface_default.xml, applied on grayscale images (cv2.cvtColor).

4.4.2. Real-Time Processing Workflow

- Video Loop: The code uses a while cam.isOpened() loop to read and process each frame from the camera, ensuring continuous processing until the user presses 'q' to exit.
- Detection and Prediction:
- Faces are detected using detectMultiScale with parameters scaleFactor=1.1, minNeighbors=10, and minSize=(10, 10), similar to demo.py, allowing detection of faces at various sizes but potentially struggling with complex angles or lighting.
- The face region (ROI) is extracted, saved as an image file in the faces_dir directory (e.g., 1.jpg, 2.jpg, ...), and converted to a tensor for prediction.
- Physiological data (EDA, HR, Temp) is manually input by the user via input(), normalized using the normalize function, and combined with the image for model input.
- Output is processed using torch.softmax to obtain probabilities and torch.argmax to determine the predicted class, with accuracy displayed.
- Result Display:

- Draws a rectangle around the face using cv2.rectangle and labels the prediction ("drowsy" or "non drowsy") using cv2.putText, providing a visual interface.

Storing face images supports post-analysis or model improvement.

4.4.3. FPS Measurement (Frames Per Second)

- FPS Calculation: The code calculates FPS by measuring the time between frames (prev_frame_time and new_frame_time), with the FPS value rounded and displayed using cv2.putText. This enables real-time performance monitoring.
- Observed Performance: FPS depends on hardware (CPU/GPU), camera resolution, and the number of detected faces.

5.4.4. Observations on Real-Time Testing

Advantages:

- Real-Time Camera Input: Using a webcam instead of a video file is more suitable for real-time driver monitoring or tracking.
- FPS Measurement: Calculating and displaying FPS (unlike demo.py where it was commented out) aids performance evaluation, critical for user experience optimization.
- Manual Physiological Data Integration: Allows users to input physiological data (EDA, HR, Temp), increasing flexibility for real-world testing.
- Data Storage: Saving face images to faces_dir supports post-analysis or model improvement.
- Visual Interface: Displaying predictions and FPS directly on frames makes results easy to track.

Limitations:

- Hardware-Dependent Performance: FPS may be low on weaker devices or when processing multiple faces, due to the heavy ViT model and Haar Cascade.
- Static Physiological Data: Manually entered EDA, HR, and Temp values remain fixed throughout the session, not reflecting real-time changes, potentially reducing long-term accuracy.
- Limited Face Detection: haarcascade_frontalface_default.xml may fail in low lighting, unusual angles, or with rapid movements, affecting reliability.
- Basic Input Error Handling: If users enter incorrect or invalid values (outside the normalized range), the code only provides basic error messages, needing a more user-friendly interface.

4.5 Discussion of Results

Advantages:

- Multi-modal integration (images and physiological data) enhances reliability.
- High performance with ~99% accuracy, particularly strong for the Fatigue class.
- Capable of real-time video processing.

Limitations:

- FPS Optimization: Techniques like model pruning or quantization could improve video processing speed.
- Lack of Augmentation: Not applying augmentation may reduce generalization to new data.
- Incomplete Training Details: Missing information on epochs, learning rate, or loss function makes the training strategy partially unclear.
- Static Physiological Data: Fixed physiological data does not reflect real-time changes.

Conclusion and Recommendations:

The training strategy in the notebook focuses on building and evaluating a multi-modal ViT model with high performance, particularly suitable for drowsiness detection in driving. To improve, consider:

- Adding augmentation to enhance generalization.
- Providing detailed training information (e.g., epochs, optimizer, scheduler).
- Integrating dynamic physiological data to reflect real-time states.

CHAPTER 5.
ETHICAL IMPLICATIONS, SOCIETAL IMPACT AND FINAL CONCLUSIONS

5.1 Ethical & Social impact

The deployment of a real-time driver monitoring system involving facial data and health indicator raises significant ethical and societal concerns. Central to these is the obligation to protect user privacy and autonomy. The system collects and processes sensitive data streams—specifically, facial imagery and heart rate signals—that could be exploited if misused or inadequately secured. As such, any implementation must enforce strict data governance protocols, including encrypted transmission, local-only processing (where feasible), and explicit user consent before data collection or inference.

In addition to technical safeguards, developers must consider broader implications such as data ownership, long-term storage policies, and the psychological impact of continuous surveillance on drivers. Transparent communication regarding how data is used, who has access to it, and under what conditions is essential to maintaining public trust and regulatory compliance. The system's design should prioritize not only effectiveness but ethical alignment with principles of fairness, transparency, and accountability.

5.2 Final conclusion

This project demonstrates how Computational Thinking can be effectively employed to design a robust, real-time driver alertness recognition system based on multimodal input. By combining deep learning techniques—specifically, a Vision Transformer for facial feature extraction—with physiological signal processing, we developed a lightweight and responsive architecture capable of per-frame binary classification under tight performance constraints.

The system currently achieves high classification accuracy with sub-second inference time using minimal input modalities. However, real-world deployment presents additional challenges that warrant further investigation. These include robustness to poor lighting, motion blur due to road vibration, and expanded support for alerting mechanisms (e.g., audio or haptic feedback). Moreover, increasing the diversity and size of the training dataset will be critical for improving generalization across drivers and driving conditions.

Ultimately, this work lays a strong foundation for scalable and ethical driver monitoring systems, while demonstrating the practical power of Computational Thinking as a problem-solving paradigm in applied AI.

DEMO

Link Github: https://github.com/Nh-ng-Ng-i-Kh-n-Kh-1-0/CS117.P21/projects?query=is%3Aopen+is%3Atemplate
Link Demo: https://drive.google.com/drive/u/1/folders/1o4rulWHGwblXRTWTNQjZmbCTyVzerYjN

TASK ASSIGNMENT BOARD

AUTHORS	TASKS	SCORE
Truong Hoang Thanh An (23520032)	• Group leader • coordinated meetings and timeline • led system architecture design; wrote CT section and final report integration.	100
Le Ngoc Thanh (23521443)	• Designed and implemented heart rate processing module • contributed to evaluation and metrics analysis.	100
Nguyen Xuan An (23520023)	• Developed facial image preprocessing and ViT-based feature extraction pipeline • supported visualization for decomposition tree and demo.	100
Vu Viet Hoang (23520548)	• Handled data preparation and HR synchronization • Created custom dataset; supported experimental setup and testing.	100
Mai Thai Binh (23520158)	• Led real-time demo implementation • handled video input integration and system deployment test • contributed to ethical discussion.	100

BIBLIOGRAPHY

[1] driver drowsiness detection system dataset.
Link: https://www.kaggle.com/datasets/amaldev007/driver-drowsiness-detection-system-dataset
[2] wesad: stress and affect detection via wearable devices
Link: https://uni-siegen.sciebo.de/s/HGdUkoNlW1Ub0Gx/download
[3] Dosovitskiy et al. (2021) – *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. Link: https://arxiv.org/abs/2010.11929*
[4] Baltrušaitis, A., Ahuja, C., & Morency, L.-P. (2019) – *Multimodal Machine Learning: A Survey and Taxonomy. Link: https://doi.org/10.1109/TPAMI.2018.2798607*
[5] Wing, J.M. (2006) – *Computational Thinking Link: https://doi.org/10.1145/1118178.1118215*
[6] Shute, V.J., Sun, C., & Asbell-Clarke, J. (2017) – *Demystifying Computational Thinking. Link: https://doi.org/10.1016/j.edurev.2017.09.003*
[7] Vural, E., Cetin, M., & Littlewort, G. et al. (2009) – *Drowsy Driver Detection Through Facial Movement Analysis. Link: https://doi.org/10.1109/ICPR.2008.4761271*
[8] Chowdhury, M.E.H. et al. (2021) – *Wearable Real-Time Drowsiness Detection Using CNN and HRV Features. Link: https://doi.org/10.3390/s21010057*