**School of Computer Science and Engineering**

**Fall 2025-26** **Course Design Project (**Phase III)

**Program Name & Branch:** B.Tech. (CSE) & All Specializations

**Course Name & Code:** BCSE332L Deep Learning

**Faculty Name (s):** Chellatamilan T

## 1. Project Overview

**Project Title: Real-Time Cognitive State Classification Using Facial Landmark Dynamics and LSTM Networks in Safety-Critical Environments**

**Prepared by:**
**Reg No:** 22BDS0381 Name: Sri Ganesan
**Reg No:** 22BDS0155 Name: R. Priyadarshini

## 2. Problem statement

Safety-critical environments such as aerospace operations, military command systems, and mission-critical industrial settings demand continuous human vigilance and precise decision-making. Even brief lapses in cognitive performance—arising from fatigue, stress, distraction, or overload—can significantly compromise safety and operational integrity. Traditional approaches to monitoring cognitive states often depend on EEG or physiological sensors, which, while accurate, are intrusive, expensive, and impractical for real-world continuous use. To address these limitations, this project proposes a real-time, vision-based cognitive state classification framework that leverages facial landmark dynamics as a non-intrusive indicator of human mental states. By capturing temporal variations in eye and facial movements through standard webcams, the system extracts structured landmark sequences that encode subtle behavioral cues linked to cognitive conditions. These sequences are then modeled using a Long Short-Term Memory (LSTM) neural network, which learns to classify users into one of several cognitive states—alert, fatigued, stressed, distracted, or neutral—in real time. The proposed system aims to provide a cost-effective, scalable, and non-intrusive alternative to biosensor-based monitoring, enabling continuous assessment of operator alertness and cognitive well-being in safety-critical environments.

## 3. Objectives

Objective 1:
To design and implement a non-intrusive, real-time cognitive state classification system that utilizes eye behavior and facial landmark dynamics captured through a standard webcam, eliminating the need for specialized biosensors or wearable devices.
Objective 2:
To train, optimize, and evaluate a sequence-based Long Short-Term Memory (LSTM) model on temporal eye and facial landmark data for accurately identifying cognitive states—such as

alertness, fatigue, stress, distraction, and neutrality—within safety-critical decision-making environments.

## 4. Data Requirements

1. Labeled dataset:

A csv file with varied facial features including: blink_rate, avg_EAR, gaze_x, gaze_y, brow_raise_freq, brow_furrow_intensity, mouth_aspect_ratio, lip_corner_disp, speaking_prob, jaw_variance, mouth_open_duration, pitch, yaw, window_start_time, window_end_time, cognitive_state.

Labels: Each tuple is labeled with the corresponding cognitive state. These labels serve as the "ground truth" for the model to learn from. Common cognitive state labels in safety-critical contexts include: 1. Stressed, 2. Fatigued, 3. Alert, 4. Neutral, 5. Distracted

2. Subject Diversity:

To ensure the model generalizes well to different individuals, the training data should include a diverse range of participants. This includes variations in:

Age and Gender: Different age groups and genders can exhibit subtle differences in eye behavior.
Ethnicity: Facial structures and eye shapes can vary across ethnicities, which can affect landmark detection.
Facial Features: The presence of glasses, beards, or other features that might partially obscure the face should be included in the dataset to make the model more robust.

3. Environmental and Contextual Variations:

The training data should mirror the real-world conditions of the safety-critical environment where the system will be deployed. This includes:

Lighting Conditions: A mix of good and poor lighting, as well as changing light conditions (e.g., driving from a sunny area into a tunnel).
Head Pose and Gaze Direction: People naturally move their heads and shift their gaze. The training data must include various head poses and gaze angles.
Vibrations and Movement: In environments like driving or piloting, there will be vibrations and sudden movements. The data should reflect these to prevent the model from being thrown off by such artifacts.

## 5. Model requirements

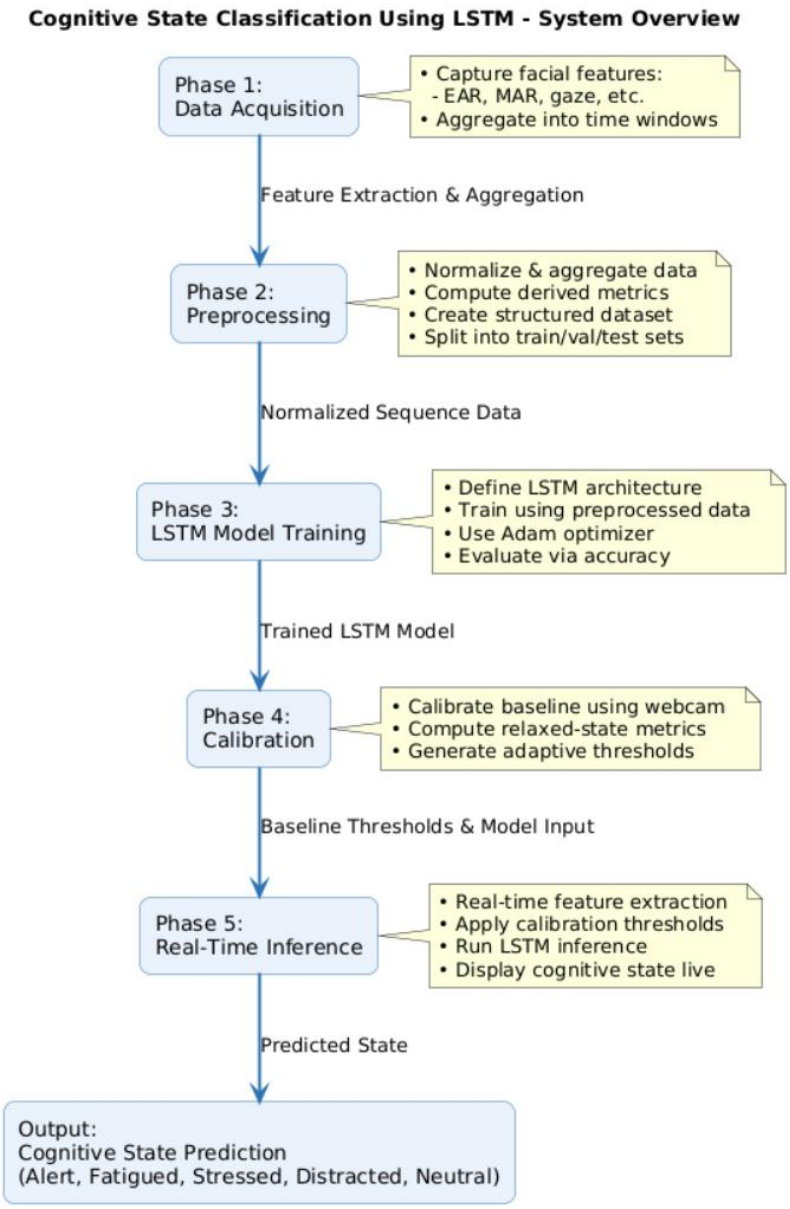
The proposed system requires a model capable of understanding temporal dependencies in multimodal facial features and differentiating between subtle behavioral cues linked to human cognitive states. As the target application operates in safety-critical domains, the model must balance accuracy, real-time responsiveness, and computational efficiency.
Functional Requirements

- Input Format: Sequential feature vectors derived from facial landmarks (eyes, eyebrows, lips, jaw, head pose) captured at frame level and aggregated over short temporal windows (typically 10 s).

- Model Architecture:
    - 1–3 stacked Long Short-Term Memory (LSTM) layers to capture time-based dependencies and micro-expressive variations.
    - Dropout layers between LSTM blocks to mitigate overfitting.
    - Dense output layer with softmax activation representing the five target cognitive states: *Alert, Fatigued, Stressed, Distracted, and Neutral*.
- Training Data: Balanced and labeled temporal sequences obtained after preprocessing and normalization.
- Performance Metrics: Overall classification accuracy, F1-score, and confusion matrix for class-wise performance analysis.
- Real-Time Capability: The model should perform inference within 100–150 ms per frame window to ensure real-time responsiveness on standard CPU hardware.
- Scalability: The architecture should remain lightweight enough for deployment in embedded or field-ready systems (e.g., laptops, onboard mission consoles).

---

## 6. Model Architecture



**Phase 1: Data Acquisition**

In this phase, multimodal facial and behavioral features are captured from each video frame or aggregated over defined time windows to form the foundation of the cognitive state recognition dataset. The eye-related features include the Eye Aspect Ratio (EAR) representing eye openness, blink count per second, average blink duration, Inter-Blink Interval (IBI), PERCLOS (Percentage of Eye Closure) as an indicator of fatigue, and gaze position coordinates (gaze_x, gaze_y) for attention tracking. Eyebrow features such as brow raise frequency and furrow intensity are recorded to reflect emotional and cognitive tension cues. Lip and mouth features include the Mouth Aspect Ratio (MAR) to measure mouth openness, lip corner displacement to assess expressiveness or engagement, and speaking probability to estimate speech activity. Jaw-related metrics, including jaw variance and mouth open duration, are captured to indicate clenching, stress, or speaking intensity. Additionally, head pose features such as pitch and yaw are extracted to determine orientation and attention level.

Each data sample corresponds to a defined time window (typically 10 seconds), during which frame-level features are aggregated using statistical operations such as mean, variance, and count to represent consistent temporal behavior. The dataset also includes temporal metadata—window start and end times—to maintain chronological coherence. The final labeled dataset targets five cognitive state classes: Alert, Fatigued, Stressed, Distracted, and Neutral. This structured and comprehensive feature set provides the necessary temporal and behavioral inputs for training sequence-based models such as Long Short-Term Memory (LSTM) networks for full-face cognitive state classification.

**Phase 2: Preprocessing**

In this phase, the collected frame-level features are preprocessed to ensure consistency and suitability for temporal modeling. First, all feature values—such as Eye Aspect Ratio (EAR), Mouth Aspect Ratio (MAR), and gaze coordinates—are normalized to a common scale (either in the range [0, 1] or standardized using z-scores) to remove bias due to varying measurement scales. The normalized data is then aggregated into fixed-size temporal windows (for example, 10 seconds) to capture short-term behavioral dynamics. Within each window, several derived metrics are computed: for eye behavior, parameters such as mean EAR, blink rate, blink duration, PERCLOS (Percentage of Eye Closure), and inter-blink interval (IBI) are extracted; for eyebrows, raise frequency and furrow intensity are calculated; for lips, MAR, lip corner displacement, and speaking probability are estimated; for jaw movement, jaw variance and mouth-open duration are derived; and for head pose, pitch and yaw values are included. The resulting window-level features are stored in a structured format (CSV or HDF5) suitable for LSTM input. Finally, the dataset is divided into training, validation, and test subsets following a standard 70:15:15 split to enable robust model development and evaluation.

**Phase 3 – LSTM Model**

In this phase, a Long Short-Term Memory (LSTM) neural network is designed and trained to classify cognitive states based on the preprocessed temporal feature sequences. The model input consists of sequential feature vectors corresponding to fixed-length time windows (typically 10-second rolling segments) generated during preprocessing. The core architecture comprises one to three stacked LSTM layers, which effectively capture temporal dependencies and dynamic behavioral patterns across time. Dropout layers may be incorporated between LSTM layers to prevent overfitting and enhance generalization. The final output layer contains five neurons activated by a softmax function, corresponding to the five cognitive states—Alert, Fatigued, Stressed, Distracted, and Neutral.

The model is trained using the preprocessed and labeled sequences from Phase 2. Categorical cross-entropy is employed as the loss function, while adaptive optimizers such as Adam or RMSProp are used to ensure efficient gradient updates. Model performance is evaluated using metrics such as overall accuracy and class-wise F1-scores to assess both general and per-category predictive capability. Once trained, the finalized LSTM model is saved and prepared for deployment in Phase 5, where it will be used for real-time inference and cognitive state classification.

**Phase 4 – Calibration**

The calibration phase establishes individualized baseline metrics to account for user-specific variations in facial geometry, expression dynamics, and lighting conditions. Initially, any required packages for visualization and interactive control—such as webcam access libraries or display widgets—are installed and configured. The system then prompts the user for permission to access the webcam and records a short video segment, typically 5–10 seconds, under relaxed and neutral conditions to represent the user's baseline state.

From this calibration session, baseline metrics are computed across all facial regions. For the eyes, parameters such as mean Eye Aspect Ratio (EAR), blink rate, PERCLOS (Percentage of Eye Closure), and Inter-Blink Interval (IBI) are extracted. Eyebrow metrics include average furrow intensity and raise frequency, while lip-related metrics encompass the Mouth Aspect Ratio (MAR), lip corner displacement, and speaking probability. Jaw-based measurements, including jaw variance and mouth open duration, are also calculated. Head pose parameters—pitch and yaw—are analyzed to capture natural orientation tendencies.

These baseline statistics serve as personalized reference values for dynamic thresholding during real-time feature normalization and anomaly detection. Additionally, a live webcam preview is provided throughout the calibration process, allowing the user to adjust their position, posture, or ambient lighting to ensure accurate and stable baseline measurement.

**Phase 5 – Real-Time Feature Extraction & LSTM Inference**

In this phase, the trained LSTM model from Phase 3 is deployed for real-time cognitive state classification using live webcam input. The process begins by loading the pre-trained model and initializing a set of helper functions responsible for facial landmark detection (using MediaPipe Holistic), frame-wise feature computation across all facial regions, and temporal aggregation of these features into a rolling buffer representing the most recent 10-second interval. For environments such as Google Colab, a JavaScript snippet is executed to create and manage a video element that enables live webcam capture.

During inference, calibration thresholds obtained from Phase 4 are applied to normalize the live-streamed features, ensuring that predictions remain consistent with the user's baseline conditions. As frames are captured continuously, the system extracts relevant features (eyes, eyebrows, lips, jaw, and head pose), updates the rolling window buffer, and passes the aggregated sequence to the LSTM model for real-time prediction. The model outputs the most probable cognitive state which is displayed instantly to the user. The process includes safe camera handling, ensuring the webcam stream is properly released and closed once the session ends, maintaining both system stability and user privacy.

---

## 7. Hyperparameter Tuning

1. LSTM Network Architecture

• Number of LSTM Layers:

The depth of the LSTM network determines its ability to model complex temporal dependencies in facial behavior.

- Range: Typically, 1–4 layers
- Recommended for this project: Start with 2 LSTM layers to balance model complexity and computational efficiency.
- Impact: A shallow network (1 layer) may underfit and fail to capture nuanced cognitive transitions (e.g., subtle shifts from alertness to fatigue), while a deeper one (3–4 layers) may increase latency and risk of overfitting due to redundant temporal abstraction.

• Number of LSTM Units (Neurons) per Layer:
Defines the representational capacity of each LSTM layer.

- Range: 32, 64, 128, or 256 units
- Recommended: Begin with 64 units in the first layer and 128 in the second, to allow hierarchical learning of short- and long-term dependencies.
- Impact: More units capture richer behavioral dynamics (e.g., blink duration trends, gaze shifts), but they also raise computational cost. The optimal configuration depends on the diversity and temporal complexity of the input feature set.

2. Training and Optimization
• Learning Rate:
Controls how much the model's weights are updated at each step.

- Range: 0.01 to 0.0001
- Recommended: 0.001 (with a learning rate scheduler to reduce on plateau)
- Impact: A high rate may cause divergence, while a low one slows convergence. Adaptive scheduling ensures stable learning and prevents oscillations in loss.

• Batch Size:
Number of sequences processed per training iteration.

- Range: 16, 32, 64, or 128
- Recommended: Batch size of 32 for balanced speed and generalization.
- Impact: Larger batches stabilize gradients but risk poorer minima; smaller batches improve generalization but increase noise.

• Number of Epochs:
Total passes over the entire training dataset.

- Range: 50 to 500 epochs
- Recommended: Train for 100–150 epochs with early stopping (monitoring validation loss).
- Impact: Early stopping prevents overfitting by halting training once performance saturates.

• Optimizer:
The algorithm that updates weights during backpropagation.

- Options: Adam, RMSprop, or SGD with momentum
- Recommended: Adam optimizer, for its adaptive learning rate and stable convergence on time-series tasks.

3. Regularization to Prevent Overfitting
• Dropout Rate:
Fraction of neurons randomly deactivated during training.

- Range: 0.2 to 0.5
- Recommended: 0.3 dropout after each LSTM and dense layer.
- Impact: Dropout discourages the network from over-relying on specific neurons, improving generalization across diverse face and lighting conditions.

• Recurrent Dropout:
Applies dropout to the recurrent connections within the LSTM.

- Range: 0.2 to 0.5
- Recommended: 0.2 recurrent dropout to stabilize temporal learning.
- Impact: Helps prevent memorization of noise in temporal dependencies, particularly beneficial in facial feature streams with minor motion jitter.

4. Input Data Structure
• Sequence Length (Timesteps):
Defines the number of consecutive frames the LSTM observes to predict a cognitive state.

- Range: 30–90 frames (1–3 seconds at 30 FPS)
- Recommended: 60-frame (2-second) sequences aggregated using the rolling window approach.
- Impact: Longer sequences capture more behavioral context (e.g., gradual fatigue), but increase latency and memory load. Shorter ones offer faster inference but might miss extended temporal cues.

Each sequence consists of aggregated and normalized features (EAR, MAR, gaze coordinates, PERCLOS, pitch, yaw, etc.) as derived in Phase 2. The resulting temporal embeddings form the input to the LSTM model for real-time cognitive state prediction.

---

## 8. Functional Requirements

The system must be capable of capturing and processing real-time facial and head movement data through a webcam to assess a user's cognitive state. It should extract frame-wise features such as Eye Aspect Ratio (EAR), Mouth Aspect Ratio (MAR), gaze position, eyebrow activity, jaw movement, and head pose using MediaPipe-based landmark detection. The preprocessing module must normalize and aggregate these features into fixed time windows (e.g., 10 seconds) and compute derived metrics like blink rate, PERCLOS, lip displacement, and pitch/yaw orientation. The system must support model training using a Long Short-Term Memory (LSTM) network that accepts sequential input, learns temporal dependencies, and outputs one of five cognitive states — Alert, Fatigued, Stressed, Distracted, or Neutral. The calibration module should establish baseline thresholds under relaxed conditions, and the real-time inference module must continuously extract live features, apply calibration adjustments, run predictions through the trained LSTM, and display the inferred cognitive state dynamically on screen.

---

## 9. Non-Functional Requirements

The system should operate in real time with minimal latency (preferably under 200 milliseconds per frame) and maintain a high accuracy and reliability in classification across varying lighting and facial conditions. It must ensure consistent performance across different hardware configurations and be resource-efficient enough to run on standard consumer-grade devices or Google Colab environments. The design should prioritize scalability — allowing for model retraining with additional data — and modularity, so that each phase (acquisition, preprocessing, modeling, calibration, inference) can be independently updated. The user interface should be intuitive, providing clear calibration guidance and responsive visual feedback during inference. Data privacy and security must be maintained by ensuring that webcam feeds and extracted features are used solely for local computation and not stored or transmitted externally without consent.

---

## 10. Visualization of Results

Phase 1: Data acquisition

```
import pandas as pd

# Load the dataset
df = pd.read_csv("cognitive_states_dataset.csv")

# Print the first 5 rows as tuples
for row in df.head(5).itertuples(index=False, name=None):
    print(row)
```

```
(0.3342153372601146, 0.0257424097472162, 0.3410478728306235, 0.8033045735180335, 0.1631877792411288, 0.2220600213968829, 0.45550
(15.378328914148671, 0.2559518055153246, 0.1514323143699633, 0.4774161894803537, 0.6018067339454473, 0.5817366474211867, 0.40146
(0.415786062069183, 0.6010826653932156, 0.7098047132723719, 0.9505920130759384, 0.1471605702056861, 0.2054782123528669, 0.52081
(0.0641954859978143, 0.5379230312678858, 0.4788631183600459, 0.3874543535702994, 0.721909573992304, 0.8106657734835124, 0.41189
(0.9915942356641668, 0.294039710039999, 0.7102433419272591, 0.1153998582808282, 0.1801856620272098, 0.3200750173276079, 0.36765
```

Phase 2: Preprocessing

```
Preprocessing complete.
Training samples: 70
Validation samples: 15
Test samples: 15
```

Phase 3: LSTM model

```
Epoch 50/50
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.
WARNING:absl:You are saving your model as an HDF5 fi
Test Accuracy: 80.00%
LSTM model saved as lstm_cognitive_state_model.h5
```

Phase 4: Calibration

```
This program will access your webcam to collect baseline facial metrics. Allow? (y/n): y
Capture each frame by clicking the 'Capture' button in the video window.
Captured frame 1/5
Captured frame 2/5
Captured frame 3/5
Captured frame 4/5
Captured frame 5/5

Baseline calibration complete:
avg_EAR: 0.2815
std_EAR: 0.0147
avg_brow: 77.3738
std_brow: 0.6154
avg_lip: 9.2975
std_lip: 0.4924
avg_jaw: 36.8231
std_jaw: 1.2997
avg_head_pose: -2.0013
std_head_pose: 0.3761
EAR_closed_threshold: 0.1971
```

Phase 5: Real-Time Feature Extraction & LSTM Inference

```
Starting live 60-second capture...
[14s] Cognitive state: fatigued (Confidence: 0.60)
[20s] Cognitive state: stressed (Confidence: 0.88)
[30s] Cognitive state: distracted (Confidence: 0.68)
[40s] Cognitive state: fatigued (Confidence: 0.59)
[50s] Cognitive state: stressed (Confidence: 1.00)
[60s] Cognitive state: stressed (Confidence: 1.00)
Capture finished. Predictions:
Time 14s: fatigued (0.60)
Time 20s: stressed (0.88)
Time 30s: distracted (0.68)
Time 40s: fatigued (0.59)
Time 50s: stressed (1.00)
Time 60s: stressed (1.00)
```

---

## 11. Tools and frameworks

The project integrates a combination of computer vision, deep learning, and real-time processing tools to ensure robustness and accessibility.

Development and Modeling

- Python 3.x – Primary language for all phases, offering compatibility with machine learning and computer vision libraries.
- TensorFlow / Keras – For designing, training, and deploying the LSTM network with sequential modeling capabilities.
- NumPy & Pandas – For handling numerical arrays, feature extraction, and dataset structuring.
- Scikit-learn – Used for dataset splitting, normalization, and performance metric computation.

Computer Vision and Feature Extraction

- MediaPipe Holistic – For real-time detection of facial landmarks, including eyes, lips, eyebrows, and head pose.
- OpenCV – For video stream handling, frame capture, and feature computation (EAR, MAR, PERCLOS, etc.).

Data Storage and Visualization

- CSV formats – For structured storage of time-windowed feature sequences.
- Matplotlib – For visualizing trends, calibration data, and confusion matrices.
- Google Colab – For code execution, real-time webcam integration, and demonstration.

## 12. Risk assessment

| Risk Category | Description | Impact | Mitigation Strategy |
|---|---|---|---|
| Data Variability | Variations in lighting, camera quality, facial geometry, and head movement may cause inconsistent feature extraction. | High | Incorporate Phase 4 calibration, adaptive thresholding, and brightness normalization; train on diverse lighting and demographic conditions. |

| | | | |
|---|---|---|---|
| Model Overfitting | The LSTM may overfit to specific subjects or scenarios, reducing generalization. | Medium | Apply dropout, early stopping, and cross-validation; maintain a balanced dataset. |
| Latency and Real-Time Constraints | Delays in feature extraction or model inference could hinder real-time usability. | Medium | Optimize preprocessing pipeline; use lightweight models and buffer-based rolling windows. |
| Privacy Concerns | Continuous webcam monitoring may raise privacy and data security issues. | High | Ensure on-device processing without cloud upload; anonymize or delete frames post-inference. |
| Incorrect Classification | Misclassification of cognitive state may lead to inappropriate operator feedback or decision bias. | High | Include confidence thresholds and fallback "uncertain" class; provide explainable model output. |
| Hardware Dependence | System performance may vary with different hardware specifications. | Medium | Benchmark across multiple setups; specify minimum hardware requirements. |
| Ethical and Human Factors | Misinterpretation of model results could impact personnel evaluation or safety decisions. | High | Frame outputs as assistive indicators, not as deterministic judgments; maintain human-in-the-loop supervision. |

---

## 13. Timeline and milestones

| Week | Dates | Phase / Activity | Key Milestones & Deliverables |
|---|---|---|---|
| Week 1 | Jul 15 – Jul 21 | Phase 1: Data Acquisition Setup | - Install and test dependencies (MediaPipe, OpenCV, TensorFlow, etc.) - Implement webcam-based data capture pipeline. - Define and compute eye, mouth, eyebrow, jaw, and head pose features. - Create initial feature logging and storage format (CSV/HDF5). |
| Week 2 | Jul 22 – Jul 28 | Phase 1: Dataset Construction | - Collect multi-user sample data under controlled conditions. - Aggregate features into 10-sec temporal windows. - Label dataset into 5 cognitive states (Alert, Fatigued, Stressed, Distracted, Neutral). Milestone 1: Structured, labeled dataset ready. |
| Week 3 | Jul 29 – Aug 4 | Phase 2: Preprocessing and Feature Normalization | - Normalize features (z-score / [0,1] scaling). - Compute statistical aggregations (mean, variance, counts). - Split data into train/validation/test (70/15/15). - Validate dataset quality and balance. |
| Week 4 | Aug 5 – Aug 11 | Phase 3: LSTM Model Design | - Build and tune LSTM architecture (1–3 layers + Dropout). - Define model training parameters (Adam optimizer, categorical cross-entropy). - Conduct initial training runs. |
| Week 5 | Aug 12 – Aug 18 | Phase 3: Model Training & Evaluation | - Train final LSTM model on full dataset. - Evaluate accuracy, precision, recall, and F1-score. - Save trained model and performance report. Milestone 2: Trained & validated LSTM model. |

| | | | |
|---|---|---|---|
| Week 6 | Aug 19 – Aug 25 | Phase 4: Calibration Module Development | - Implement baseline capture (neutral video segment). - Extract per-user baseline metrics (EAR, MAR, PERCLOS, etc.). - Integrate adaptive thresholding for live normalization. - Enable real-time visualization preview. |
| Week 7 | Aug 26 – Sep 1 | Phase 5: Real-Time Inference Integration | - Connect live webcam stream with trained LSTM model. - Implement rolling 10-sec feature buffer. - Enable real-time prediction and on-screen display. Milestone 3: End-to-end real-time inference working prototype. |
| Week 8 | Sep 2 – Sep 8 | Testing & Risk Validation | - Conduct system validation under varied lighting, subjects, and conditions. - Evaluate latency, stability, and misclassification risk. - Apply mitigation strategies (thresholding, smoothing). |
| Week 9 | Sep 9 – Sep 15 | Final Integration & Documentation | - Prepare final report and presentation. - Consolidate results (confusion matrix, F1-scores, demo video). - Document risks, limitations, and future improvements. Milestone 4: Final deliverables submission. |

---

**14. Presentation Link**

Video Link: https://drive.google.com/file/d/1DFgfoO_gnqYwve4aF-yBrk09huZsPXvj/view?usp=sharing

---

**15. Student Information**


| | |
|---|---|
| Name: Sri Ganesan | Name: R Priyadarshini |
| Reg. Number: 22BDS0381 | Reg. Number: 22BDS0155 |
| Department/Course: School of computer science engineering / Deep learning | Department/Course: School of computer science engineering / Deep learning |
| Email ID: sriganesaniyer@gmail.com | Email ID: priyadarshini10012005@gmail.com |