DLSU Research Congress 2025
De La Salle University, Manila, Philippines
June 26 to 28, 2025

Driver Attention Monitoring System with AI-Based Assistant for Fleet Operations

Michael Aringo[1], Henson Lee[1] , Gabriel Ocampo[1] , Jared Ong[1] , and Melvin Cabatuan[1]
[1] *De La Salle University, Gokongwei College of Engineering*
**Corresponding author: michael_aringo@dlsu.edu.ph*

Abstract: Driver inattention is a primary contributor to vehicular accidents globally, frequently arising from drowsiness or distractions such as phone usage. Traditional in-vehicle safety systems often concentrate on a single indicator (e.g., eye closure) and fail to adapt to the complex dynamics of real-world driving. This paper presents a comprehensive driver monitoring system for fleet operations that integrates an ESP32 camera module (ESP32-CAM) equipped with infrared (IR) LEDs, advanced Convolutional Neural Network (CNN) architectures, Global Positioning System (GPS) tracking, and an AI-based assistant powered by a Large Language Model (LLM). The system captures real-time driver facial data in varied lighting scenarios, classifies multiple inattentive states (drowsy, focused, yawning, and using a phone), and provides context-aware vocal prompts when risky behaviors are detected. Additionally, the driver's status and GPS coordinates are logged in a cloud database, enabling fleet managers to track vehicle location and driver attention. Six CNN architectures, chosen specifically for their computational efficiency and suitability for real-time mobile deployment, were evaluated on a dataset of 3,663 images. MobileViT v2 emerged as the top performer, achieving approximately 94.8% accuracy while maintaining a lightweight structure suitable for embedded systems and mobile applications in fleet environments. The results underscore the potential of integrating lightweight yet robust CNN models with real-time AI-driven interventions and location logging to bolster fleet safety and operational efficiency.

Key Words: Driver Monitoring System, CNN, ESP32, GPS, AI Assistant, Fleet Operations

1. Introduction

1.1 Background

Driver distraction and drowsiness contribute substantially to vehicular accidents worldwide. According to multiple studies, up to 90% of road incidents may be influenced by human factors (Alkinani et al., 2020). In the Philippines, driver inattention has been identified as a central contributor to collisions, with further complications arising from congested urban traffic (Castilla et al., 2023). Driver fatigue is particularly concerning for fleet operations: many drivers operate vehicles for extended hours, increasing the likelihood of drowsiness and poor decision-making (Santos & Lu, 2016). Legislative measures such as Republic Act 10913 (the Anti-Distracted Driving Act) have been enacted to combat cell phone use and other distractions. However, a broader solution is needed to detect multiple forms of inattention and give real-time feedback without intruding too heavily on the driver's routine. Emerging advanced driver-assistance technologies use sensors and artificial intelligence to detect early symptoms of driver fatigue or distraction (Albadawi et al., 2023). Yet most existing systems either focus on just one or two indicators (eye closure or yawns) or require expensive, specialized hardware.

1.2 Problem Statement

Conventional monitoring systems frequently fail to

capture the full range of inattention markers, from consistent phone use to subtle drowsiness cues like yawning. Moreover, many solutions do not adequately address challenging lighting or real-world conditions. For fleet operators, a lack of real-time insights into drivers' attention levels makes it difficult to preempt accidents. The immediate challenge is to create a robust, multi-faceted monitoring platform that can integrate seamlessly with fleet vehicles and respond in real time.

2. METHODOLOGY

This study employs a comprehensive experimental framework designed to continuously monitor driver attention in real time by integrating hardware, software, data processing algorithms, and a centralized server system. The overall structure of the system enables it to capture a driver's image, process the video feed to detect signs of inattention, and transmit the data along with corresponding Global Positioning System (GPS) information to a cloud-based server. This enables real-time monitoring and analysis by fleet operators while also providing immediate auditory feedback to the driver through an AI assistant. The following sections describe in detail each component of the system, the process of data collection and classification, and the experimental design used for evaluation.

2.1 System Architecture and Research Design

The proposed driver monitoring system is comprised of four major modules. At its core, the system integrates a hardware component, a software component, an algorithm for driver state classification, and a server component for data logging and fleet management. The hardware, consisting of an ESP32-CAM module coupled with infrared (IR) illumination, is mounted on the vehicle's dashboard to ensure an unobstructed, real-time view of the driver. The captured video is transmitted to an Android smartphone, which runs an application built using the Kivy framework. This application processes the incoming video data and uses a Convolutional Neural Network (CNN)-based algorithm to classify the driver's state into various categories, such as focused, drowsy, yawning, or using a phone. In addition to processing the video stream, the application acquires GPS data to log the driver's precise location. These data are transmitted to a cloud-hosted database that is accessible via a web portal by fleet operators, enabling them to monitor driver performance and analyze trends over time. Figure 1 (see conceptual illustration below) presents a high-level overview of the research pipeline, delineating the flow from data capture to actionable insights.

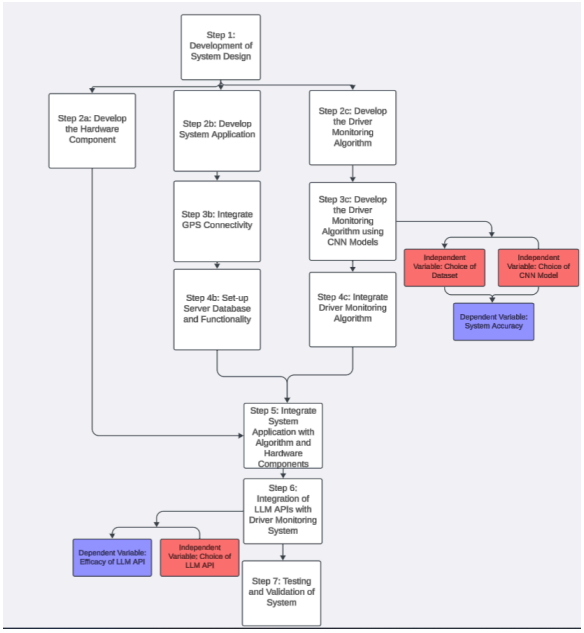

Figure 1. Conceptual Diagram of the Overall System Architecture

2.2 Hardware component

The hardware component is engineered for reliable, high-quality data capture across diverse real-world driving scenarios. It consists of 2 primary components: an ESP32-CAM-based camera module and an Android smartphone.

The camera module integrates an OV2640 camera sensor sensitive to both visible and infrared light. The OV2640's IR filter was removed so as to allow the usage of IR LEDs, allowing the system to produce crisp imagery even in low light conditions using IR light invisible to the driver and therefore non-distracting. It is housed in a compact enclosure and mounts low to the dashboard, offering an unobstructed view of the driver's face while remaining fully compliant with safety regulations such as Republic Act 10913. Power is drawn directly from the vehicle's onboard ports, ensuring continuous, stable

operation.

An Android smartphone, with built-in GPS capability, a quad-core processor, and at least 4 GB of RAM, serves as the processing unit: it wirelessly receives the live video feed from the ESP32-CAM and runs the real-time driver-monitoring application. The use of an Android smartphone to run the application allows the system to run on readily available devices with ample performance allowing the system to remain low cost while still retaining performance superior to most embedded systems

2.3 Algorithm component

The algorithm component is responsible for processing the live video stream to monitor driver attention through a two-stage process involving face detection and state classification. Initially, every frame captured by the camera is analyzed using a Haar Cascade–based face detection algorithm. This detection routine identifies the driver's face and crops the surrounding region so that only relevant facial features (such as the eyes, mouth, and overall head orientation) are forwarded for further analysis.

Once the image is cropped, it is passed to an attention monitoring algorithm that utilizes a CNN for state classification. The CNN model has been trained to recognize four distinct driver states: focused, drowsy, yawning, and using phones. In this system, the focused state indicates optimal alertness, whereas the drowsy and yawning states serve as indicators of fatigue. The state "using phone" is interpreted as a sign of distraction due to mobile device usage.

The classification process operates continuously: if no face is detected for a predetermined number of consecutive frames, the algorithm infers that the driver's face is not within the optimal field of view—indicating possible distraction or poor positioning. Conversely, when a face is consistently recognized, the algorithm processes each frame with the CNN to determine the corresponding driver state. If non-focused states (drowsy, using phone, or yawning) persist for more frames than the set threshold, the system determines that the driver is experiencing inattention and issues an appropriate warning. Similarly, consistent detection of phone use triggers an alert that the driver is distracted. When the majority of frames indicate that the driver is focused, the system remains in monitoring mode without issuing alerts.

2.4 Software & Database Component

The software and database component consists of a Kivy-based Android application integrated with a cloud-hosted database and an interactive web dashboard, collectively serving as the system's backend for fleet management. The Android application establishes a connection with the ESP32-CAM to receive the live video stream and feeds each frame to the CNN-based algorithm. As the algorithm processes the images, the application categorizes the driver's state—identifying whether the driver is drowsy, distracted, or focused—and automatically issues warnings if unsafe states persist.

The application makes use of a Python based Kivy framework that allows the app to make use of powerful and flexible Python libraries while retaining the ability to be deployed on Android devices, and make use of device functionality such as the GPS and speakers.

In addition to real-time driver status monitoring, the application collects GPS location data continuously. Both the driver's attention metrics and geographic information are transmitted securely to a remote database. This centralized repository supports a web interface that fleet managers can access. The web dashboard presents comprehensive logs of driver behavior, including real-time updates, historical performance statistics, and detailed maps illustrating the route along with specific locations and timestamps at which inattention events were detected.

Furthermore, the application incorporates an AI assistant that leverages the Google Gemini API to provide interactive, verbal feedback. When the algorithm registers a state of inattention, the AI assistant engages with the driver by issuing context-aware warnings and suggestions. This dynamic interplay between visual monitoring and conversational feedback helps maintain driver alertness while providing fleet operators with actionable insights to optimize routes and schedules for improved safety.

2.5 Test procedure

Three families of CNN architectures are evaluated for their suitability in monitoring driver attention. Specifically, the study considers two variants from each of the following families: EfficientNet (EfficientNet_lite0 and EfficientNetv2_rw_t), MobileViT (Mobilevit_xs and Mobilevitv2_100), and MobileNet (Mobilenetv3_large_100 and Mobilenetv3_small_100).

Each architecture is trained for 20 epochs on a custom dataset comprising 3,663 images captured under various driver states. The dataset is partitioned into training, validation, and test sets to ensure rigorous model evaluation and to mitigate overfitting. During training, models are continuously monitored using loss and accuracy metrics, and the best-performing epoch based on validation loss and accuracy is selected for further evaluation.

Following the initial training phase, each selected model undergoes an additional test run to assess its performance comprehensively. Evaluation metrics include test accuracy, precision, recall, and the generation of confusion matrices to reveal class-specific performance. The primary criteria for final model selection are high test accuracy and precision. The model that achieves optimal performance based on these metrics will be implemented in the final version of the driver monitoring algorithm.

Latency testing is then performed using the application on 2 devices, namely a Windows PC during program development and troubleshooting, and an Android smartphone, specifically the Samsung Galaxy Note 9 during actual implementation. This is to determine the efficacy of the application when deployed on real world environments compared to the test environment and to calibrate it as necessary.

Finally, real world integrity testing is done by deploying the system in a series of test drives on real world scenarios lasting 30 minutes to 1 hour. This is done by deploying the model on the passenger seat of the car and have the passenger simulate driving and different driving statuses so as not to endanger the researchers or other motorists.

2.6 Dataset Description

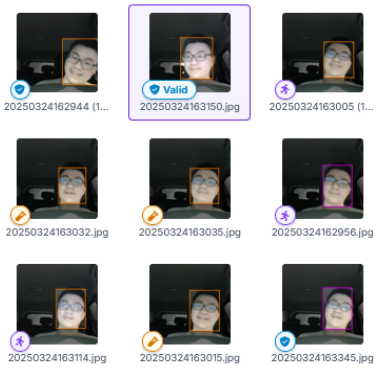

Figure 2. Sample Images from the created Driver Attention Dataset

The dataset used in this study initially comprised 3,663 labeled images of various driver attention states. These images were manually collected using the ESP32-CAM module under controlled conditions to simulate real-world scenarios. Data were captured in varying lighting environments—including daylight, nighttime, and low-light settings using IR illumination—to ensure robustness against illumination variance. The four attention states include focused, drowsy, yawning, and using a phone, with researchers themselves serving as subjects under informed consent and non-driving conditions to simulate distracted and fatigued behaviors safely.

Each image includes bounding box annotations identifying the driver's face. The icons at the lower-left corner of each image represent the dataset split assignment used during model training and evaluation. Specifically:

- The running person icon indicates the image is part of the training set.
- The shield icon designates the validation set.
- The test tube icon represents the test set.

These indicators are standardized symbols used by the Roboflow platform to visually distinguish dataset partitions in the annotation interface.

To increase dataset diversity and mitigate the

risk of overfitting, data augmentation was applied to the training set. Each image was augmented with three additional variants, resulting in a total of 9,543 images used for model training and evaluation. The final dataset was split into three subsets:

- Training set (92%): 8,820 images
- Validation set (4%): 359 images
- Test set (4%): 364 images

No preprocessing (e.g., grayscale conversion or normalization) was applied before augmentation, ensuring the model learned directly from natural RGB conditions. The following augmentation techniques were applied per training image:

- Horizontal Flip
- Rotation: Between −10° and +10°
- Shear Transformation: ±13° horizontally, ±10° vertically
- Hue Adjustment: Between −9° and +9°
- Saturation Shift: Between −25% and +25%
- Brightness Adjustment: Between −23% and +23%
- Exposure Compensation: Between −13% and +13%
- Blur: Gaussian blur up to 2.6px
- Noise Injection: Up to 1.05% of pixels

This augmentation pipeline significantly enhanced the variability of the training data, promoting better generalization during inference and increasing the model's robustness against real-world distractions such as motion blur, lighting shifts, and color distortions. The uniform application of augmentation across all classes ensured class balance and minimized performance bias during model evaluation.

3. RESULTS AND DISCUSSION

3.1 Algorithm testing results

The training and evaluation of the CNN models were conducted using a custom dataset created by the researchers. This dataset was partitioned into three distinct subsets: training, validation, and testing. The training subset facilitated the learning process, while the validation subset was used during training to monitor and fine-tune model performance. The independent testing subset was reserved exclusively for evaluating the final generalization capabilities and performance metrics of each trained model. All architectures underwent a standardized training regimen spanning 20 epochs, ensuring consistency and comparability of outcomes across different model configurations.

Table 1. Test Metrics for Evaluated CNN Architectures

Model architecture	Test accuracy	Test Loss	Validation Accuracy	Validation Loss	Best Epoch
Efficientnet_lite0	0.9260	0.3092	0.9025	0.5414	10
Efficientnet v2_rw_t	0.9479	0.2511	0.9192	0.5638	18
MobileViT_xs	0.9288	0.2941	0.9220	0.4736	14
MobileViTv 2_100	0.9479	0.2784	0.9081	0.4623	9
Mobilenetv3 _large_100	0.9288	0.3711	0.9081	0.7811	10
Mobilenetv3 _small_100	0.9452	0.2789	0.9109	0.5876	19

Table 1 summarizes the testing outcomes for each model architecture, presenting key metrics that include test accuracy, test loss, validation accuracy, and validation loss at their respective optimal epochs. Analysis of these metrics highlights three significant observations:

Firstly, all models achieved comparable accuracy results. EfficientNet Lite and MobileNetV3-Large each reached approximately 92.8% in test accuracy, while EfficientNetV2_rw_t, MobileViT_v2, MobileViT_xs, and MobileNetV3-Small achieved around 94–95% accuracy

Secondly, MobileViT-based architectures demonstrated the best overall performance. These models achieved higher accuracy and lower loss values compared to the other architectures in both testing and

validation, while also maintaining a smaller overall architecture.

Thirdly, EfficientNet-based architectures ranked second overall, with results close to or within the margin of error compared to the MobileViT models. Notably, EfficientNetV2_rw_t matched MobileViT_v2 in test accuracy. However, these models are generally larger and more complex than MobileViT, which may limit their suitability for mobile environments.

Finally, MobileNet-based architectures performed the least favorably among the models tested, exhibiting the highest loss values and only moderate accuracy.

Overall, EfficientNet and MobileViT based models outperformed the MobileNet based models. Among them, MobileViT-v2 emerged as the most effective model, achieving strong accuracy and loss metrics while maintaining a smaller model size than comparable EfficientNet models.

Table 2. Weighted average test scores for all models

Model architecture	Precision	Recall	F1- score	support
Efficientnet_lite0	0.93	0.93	0.93	365
Efficientnetv2_rw_t	0.95	0.95	0.95	365
Mobilevit_xs	0.93	0.93	0.93	365
Mobilevitv2_100	0.95	0.95	0.95	365
Mobilenetv3_large_100	0.93	0.93	0.93	365
Mobilenetv3_small_100	0.95	0.95	0.94	365

Table 2 provides details into the weighted average scores for precision, recall, and F1-score metrics obtained through testing, offering additional dimensions for model evaluation. Despite close numerical proximity among models, subtle yet consistent differences reinforce the insights derived from the accuracy and loss metrics.

EfficientNetV2_rw_t and MobileViT_v2 both achieved the highest balanced performance across precision, recall, and F1-score, underscoring their capability to correctly identify various driver states consistently. MobileNetV3_small variants closely followed these leaders, indicating solid, though slightly inferior, performance.

Conversely, MobileNetV3_large, EfficientNet Lite, and MobileViT_xs showed relatively weaker overall performance compared to MobileViT_v2 and EfficientNetV2_rw_t, aligning with earlier observations.

MobileViT_v2 demonstrates the strongest overall performance, achieving the highest accuracy and lowest loss metrics, while also utilizing a smaller, more efficient architecture. Its performance is characterized by superior accuracy, low loss, and balanced high performance across precision, recall, and F1-score metrics. This positions MobileViT_v2 as the optimal choice for implementation in the final driver attention monitoring algorithm.

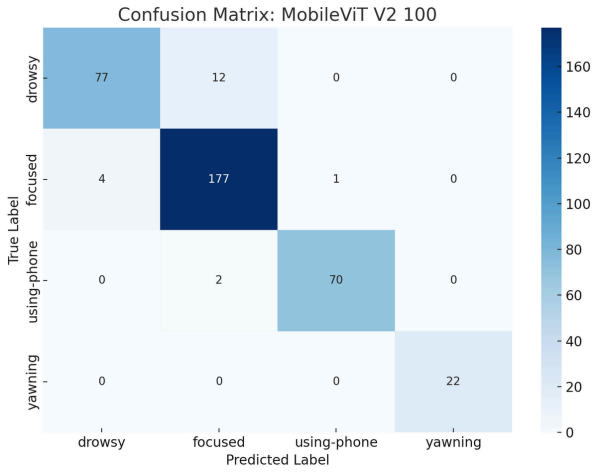

Figure 3. Confusion Matrix of MobileViT_v2

Figure 3 presents the confusion matrix derived from the classification results of the MobileViT_v2 model, which was previously identified as the top-performing architecture. The matrix offers a granular view of how the model performs across all

driver attention states: *focused*, *drowsy*, *using phone*, and *yawning*. The strongest diagonal values indicate high accuracy in classifying *focused* and *using phone* states, with 177 and 70 correctly predicted instances respectively. Notably, the *yawning* class was classified with perfect precision, while *drowsy* exhibited minor misclassification into *focused*, which may be attributed to overlapping facial expressions like half-closed eyes or slack jaw posture.

Nonetheless, the matrix validates that MobileViT_v2 reliably distinguishes between the most frequent and safety-critical behaviors, making it a strong candidate for real-time deployment in fleet vehicles.

3.2 Application

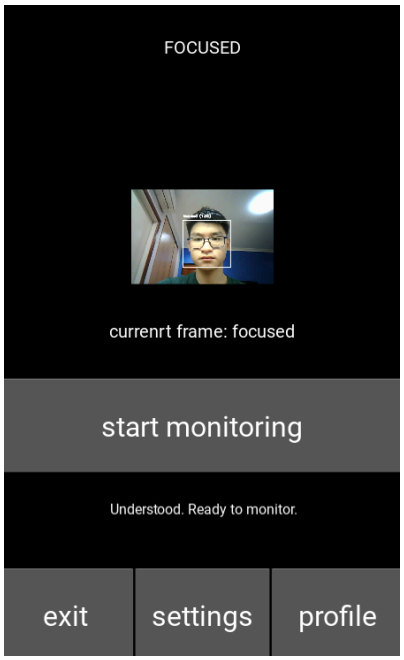

Figure 4. Screenshot of Application

Figure 4 shows a screenshot of the application developed for the software component of the system. The application utilizes a Python-based backend along with the Kivy framework for the graphical user interface (GUI). It successfully connects to the camera module and performs driver monitoring tasks reliably. Additionally, the Gemini-based assistant effectively alerts the driver when signs of drowsiness or distraction are detected.

Table 3. Test results for latency testing

Test platform	Average latency (in seconds
Laptop PC	0.0731
Galaxy Note 9	3.4645

Table 3 showcases the average latency per frame of the application running on each device tested. As can be seen, the laptop can spend less than 100 milliseconds on each frame, allowing for a framerate of up to 12 FPS, while the Android phone was only able to average roughly 3.4 seconds on each frame. This shows that latency performance is heavily dependent on hardware capabilities; as such the algorithm sensitivity was optimized to account for this latency.

The application was tested during multiple test drives between 30 minutes to 1 hour in length. This allowed the system to demonstrate its performance in real world scenarios. The system was able to function reliably and showcased robustness even in low light scenarios or areas with low signal strength. It should be noted however, that testing showed that the system encountered connectivity lag in areas with low signal strength, though the system was able to run without issues despite the lag, highlighting the integrity of the IoT-based system.

4. CONCLUSIONS

This study presents the successful development and evaluation of a real-time Driver Attention Monitoring System designed to enhance safety in fleet operations by detecting and responding to signs of driver inattention. The proposed system integrates low-cost yet effective hardware (ESP32-CAM with IR LED), mobile processing via an Android-based Kivy application, and advanced CNN architectures for visual classification. Additionally, it incorporates GPS tracking and an AI-based assistant powered by a Large Language Model for vocal feedback—creating a multi-modal solution that goes beyond traditional, single-indicator monitoring systems.

Among the six evaluated CNN models, MobileViT_v2 emerged as the most effective architecture, achieving the highest classification accuracy (94.79%) and low loss values while using a smaller, more computationally efficient model. It demonstrated superior generalization across all tested inattentive states, including focused, drowsy, yawning, and using phone, as verified through confusion matrix analysis and classification reports, underscoring the system's modular design and future extensibility.

The study additionally involved the development and evaluation of an Android application designed to operate the system. The application successfully connected to the camera module, performed driver monitoring using the implemented algorithm and utilized a Large Language Model (LLM)-based assistant to provide prompt and dynamic real-time warnings.

The application was further evaluated through extended test drives lasting approximately one hour. These tests demonstrated its ability to effectively monitor the driver and showcased the system's robust integrity under real-world conditions.

The combination of real-time face detection, behavioral classification, location logging, and AI-driven verbal warnings ensures that the system not only monitors but also actively mitigates risky behaviors on the road. Furthermore, the integration with a cloud-based dashboard empowers fleet managers with continuous visibility into driver behavior and movement, enabling data-driven decisions that can reduce accident risk, optimize scheduling, and enhance driver accountability.

In summary, this work demonstrates that an AI-powered, IoT-enabled driver monitoring system can be feasibly and effectively deployed using accessible technologies. By leveraging lightweight deep learning models and conversational AI, the system provides a scalable and intelligent solution for minimizing human error and improving road safety in commercial transport operations. Future work may include expanding dataset diversity, implementing edge optimization for broader hardware compatibility, and integrating physiological sensors to further enrich detection accuracy.

5. REFERENCES

Albadawi, Y., AlRedhaei, A., & Takruri, M. (2023). Real-time machine learning-based driver drowsiness detection using visual features. Journal of Imaging, 9(5), 91. https://doi.org/10.3390/jimaging9050091

Alkinani, M. H., Khan, M. A., & Ahmad, J. (2020). Detecting human driver inattentive and aggressive driving behavior using deep learning: Recent advances, requirements, and open challenges. International Journal of Social Science and Human Research, 3(6), 1–13. https://doi.org/10.30566/ijoas/2020.612

Castilla, J. C., Pacheco, J. L., & Sison, M. G. (2023). Occupational safety conditions of bus drivers in Metro Manila, the Philippines. Journal of Transport & Health, 28, 101471. https://doi.org/10.1016/j.jth.2022.101471

Santos, A. M., & Lu, D. (2016). Predictive variables for musculoskeletal problems in professional drivers: A systematic review. Transport Policy, 52, 93–104. https://doi.org/10.1016/j.tranpol.2016.07.001

Republic Act No. 10913, An Act Defining and Penalizing Distracted Driving. (2016). https://lawphil.net/statutes/repacts/ra2016/ra_10913_2016.html