Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Design and Implementation Stock Opname for Tire Using AI and API

Yoeka Virya Adhitthana [1], Ahmad Nurul Fajar [2], Veronica Lestari Jauw[3], Nunung Nurul Qomariyah[4]
[1]*Bina Nusantara University Jakarta, Indonesia, yoeka.adhitthana@binus.edu*
[2]*Information Systems Management Department BINUS Graduate Program- Master of Information Systems Management Bina Nusantara University Jakarta Indonesia, afajar@binus.edu*
[3] *Derpatment of Mechanical, Materials and Manufacturing Engineering University of Nottingham Malaysia Semenyih, Malaysia, veronica.jauw@nottingham.edu.my*
[4]*Computer Science Department School of Computing and Creative Arts Bina Nusantara University Jakarta, Indonesia, Nunung.qomariyah@binus.edu*

ARTICLE INFO

Received: 26 Dec 2024
Revised: 14 Feb 2025
Accepted: 22 Feb 2025

ABSTRACT

The stock opname application for tire sales must have features that help in recording, monitoring, and analyzing stock in real time. The traditional stock opname process is carried out by workers working in the warehouse using a scanner or by manually counting SKUs, but this requires careful workers so that the stock calculation is correct. The purpose of this study is to create a tire stock opname application using Computer Vision and AI (Artificial Intelligence) technology. By using tire photos, tire information can be obtained such as Brand, Model, DOT, and Size, so you only need to input the description and amount of stock. The stock opname application is built using Flutter, and is supported by Flask which acts as an API, utilizing YOLOv8 and TrOCR to extract text from the tire photos entered.

Keywords: AI, API, Stock Opname, Design, Implementation

INTRODUCTION

The stock opname application for tire sales must have features that help in recording, monitoring, and analyzing stock in real time. If you want to create a tire stock opname application with object detection, then the application can use Computer Vision and AI (Artificial Intelligence) technology to automatically detect tires through the camera. Its use in automotive stores, then it will be able to do stock count with this application. By using tire photos, the information can be obtained, so you only need to input remarks and the amount of stock, then you can also edit the amount of stock and remarks later. If the one in the photo is not a tire, then it can also be used, for example to input brake stock. The stock opname application for tire sales must have features that help in recording, monitoring, and analyzing stock in real time. If you want to create a tire stock opname application with object detection, then the application can use Computer Vision and AI (Artificial Intelligence) technology to automatically detect tires through the camera. Tire Object Detection can be done by Using a camera or smartphone to scan tire stock in a warehouse/store. The use of AI can recognize the number of tires detected in the image/video Identify the brand, size, and type of tire through the tread pattern or label on the tire.

This application will use Computer Vision to detect tires through the camera, then send the data to the Backend API for stock recording. APIs are very important in tire object detection systems as they help in communication, data processing and integration with other systems. API allows frontend (mobile/web app) to send images to backend to be processed by AI model (YOLO/OpenCV). Without API: AI model must be run directly on user device, which requires a lot of computing power. With API: Detection process is done on server/cloud, faster and more efficient. Mobile/web app takes tire image → Sends to API → Server processes & returns detection result. If a tire shop has 10 branches, the API allows all branches to use the same detection system, without having to install an AI model on each device. The API makes it easy for the tire object detection system to connect with other applications. With APIs, applications can run on multiple platforms (mobile/web) without having to rewrite code for each device. Accessible

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

from Android, iOS, and browsers and No need to build separate AI models for each platform. The API accepts image input and processes it with an AI model. The API returns detection results in the form of bounding box coordinates and classification.

Integrate the API with web or mobile applications. Host the API in the cloud and use the service for the frontend. APIs act as a bridge between AI models and the applications that use them. Tire detection applications using AI and API have many benefits, especially in the automotive, transportation, and safety industries. Here are some of the main benefits: AI can quickly recognize tires without the need for manual inspection, The detection process can be done in seconds compared to manual inspections that take longer, AI is able to analyze tire tread patterns and detect wear more accurately than humans, Reduce human error in tire inspection, especially in different lighting conditions or viewing angles, Early detection of tire wear or damage to prevent accidents on the road, Identify tire pressure and condition to avoid the risk of tire blowouts while driving, Reduce repair costs due to previously undetected tire failures, With API, tire detection systems can be integrated into mobile applications, vehicle dashboards, or fleet management systems. This architecture will consist of several main components such as Mobile/web application for users (cashiers, admins, warehouse staff), Camera for real-time tire detection, Dashboard to view stock, reports, and item status, API Server (Node.js, FastAPI, or Django REST Framework) for data communication, Database (MySQL / PostgreSQL / Firebase) to store stock data, AI model (YOLO / TensorFlow / OpenCV) for tire object detection, YOLO / OpenCV model to detect the number and type of tires, Image recognition process to recognize tire brands, sizes, and types, and Validation with OCR (Optical Character Recognition) if necessary to read the label.

According to [1], Optical character recognition (OCR) technology has revolutionized text recognition globally, enabling the conversion of text from images into a machine-readable format, thereby eliminating manual data entry. OCR has evolved to decipher handwritten content and has been supported by advances in artificial intelligence. The sidewall of the tire needs to be considered, because Understanding the sidewall markings of the tire is very important for consumers when buying bar tires. This is important because it relates to performance and safety standards [2]. While on the other hand, there is an increasing demand for mobile-based solutions that can assist users on the go for vehicle owners [3]. This mobile-based application for tire analysis can offer a portable, easy-to-use platform to quickly and precisely analyse tire information

RELATED WORKS

According to [4], a method to detect tyre defects has been done. Besides that, [5] has been proposed method to read the text of tire. According to [6], it is important to understand tire specifications to make the right purchasing decision. Understanding tire specifications will have a significant impact on tire performance and durability. However, the large gap in awareness among the public causes the purchase of inappropriate tires, which affects vehicle efficiency and market trends. Meanwhile, according to [8], the lack of consumer education efforts by tire manufacturers and regulatory bodies, exacerbates consumer ignorance. This condition will contribute to uninformed decision making. According to [8], train supervised deep learning and image processing algorithms can detect and recognize brands on tire sidewall images. Besides that, Quality control in the tire industry has leveraged AI to support transforming the tire industry, resulting in safer, more reliable and more sustainable tires [9]. Implementation of mobile application for agile environment has been implemented by [10]. According to [11] that explains that there are obstacles for companies such as missing opportunities to sell goods to hospitals due to less than optimal stock recording, so they focus on contributing by performing high-accuracy text detection and recognition to accurately recognize the number and LOT number of goods remaining in hospitals, and helping salespeople to visit hospitals less frequently but more efficiently. Its uses OpenCV and Tesseract OCR, which were chosen due to limited datasets. Related to [12], it explains that the task of recording stock requires careful human effort, but the large quantity and high SKU variation are factors for implementing an automatic stock taking system. In this paper, drones supported by Computer Vision and AI are used as a means to automatically record stock, but the methods or techniques used are not explained in more detail.

METHODS

In this study, several methodologies will be explained below :

1. **Problem Identification & Research Objectives**

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

This research aims to improve the efficiency of the stock-taking process. Traditionally, stock-taking is carried out by warehouse workers using scanning devices or manually counting SKUs in the warehouse area. The high variation in goods adds complexity to the stock-taking process, requiring meticulous workers to ensure accuracy. Therefore, this study aims to develop a system capable of capturing product information variations through image recognition and automatically recording them into the system, allowing workers to focus more on the quantity of items being counted.

2. **Literatur Review**
A literature review is conducted in this study to build a comprehensive research framework. The review includes an analysis of model algorithm development to meet specific needs, an evaluation of the integration process between APIs and fine-tuned models, and an assessment of how to integrate the applications with APIs and deploy them for real-world applications.

3. **Model and Dataset Selection**
This research focuses on extracting information from vehicle tire sidewall images, specifically the Brand, Model, DOT, and Size. Based on this objective, the Tyre Sidewall dataset is required for training the YOLOv8 algorithm. Additionally, to extract text from the detected Regions of Interest (ROI), a curved text dataset is needed for TrOCR to improve text extraction accuracy. In this study, the SCUT-CTW1500 dataset is used for fine-tuning the TrOCR algorithm.

4. **Architecture Selection**
To connect the developed model to real-world applications, a well-structured architecture is required to enable user interaction. The development process includes building a mobile application using Flutter, allowing users to interact with the system seamlessly.
To enable communication between the mobile application and the algorithmic model, a Flask-based API is developed. The API plays a crucial role in this research, handling data processing from the mobile application, including integrating with the model for image classification and text extraction.
The developed API is deployed in a containerized environment on Microsoft Azure. The database used is a PostgreSQL-based Platform as a Service (PaaS) to accelerate development. Since the models are large, they are stored in Azure Blob Storage, allowing the API to access them only when needed. Additionally, Firebase Remote Config is utilized to simplify application configuration management.

5. **Prototype Design and Implementation**
The prototype is developed as an Android application using Flutter, while the API is built using Flask, a Python-based framework. The YOLOv8 medium model is used for image classification and the Microsoft/trocr-small-printed model is used for OCR. These models undergo fine-tuning with domain-specific datasets and are then integrated into the API.
To enhance accuracy, pre-processing and post-processing steps are implemented in the API, including converting images to grayscale, applying denoising techniques, and validating output with LLM (Large Language Models). The API is deployed on Microsoft Azure using a cloud-based environment, while the mobile application is distributed via Firebase for easy deployment and updates.

RESULTS AND DISCUSSION

In this study we create system architecture and prototype of the system that can be explained below :

1. **System Architecture for Tyre2Text**

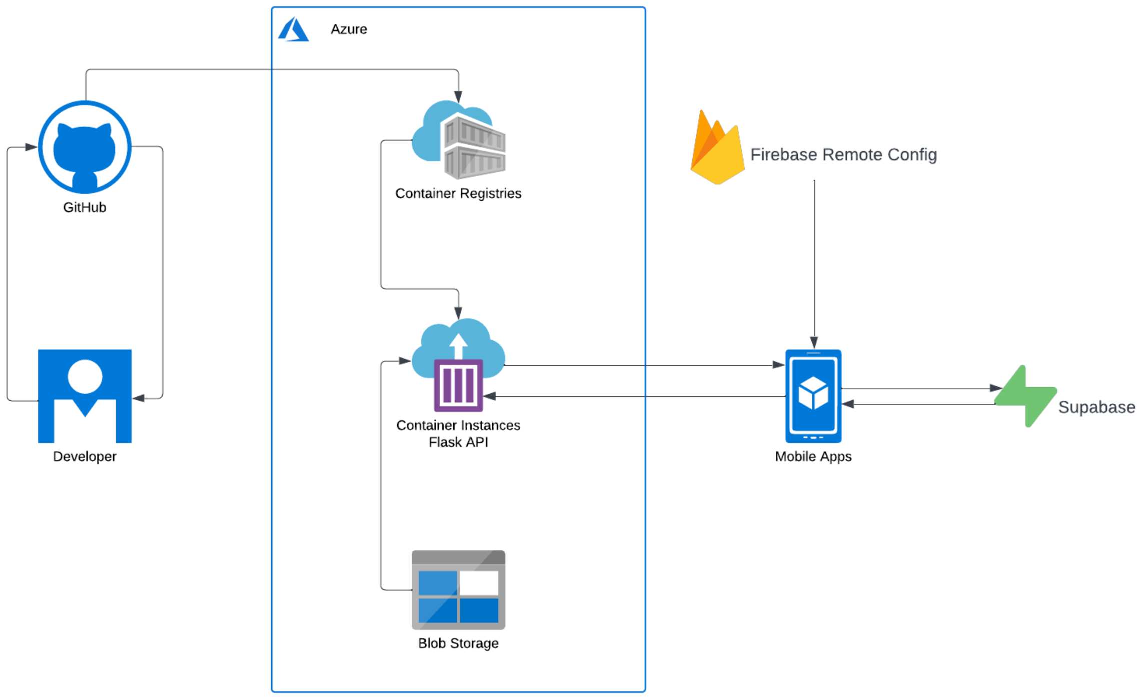
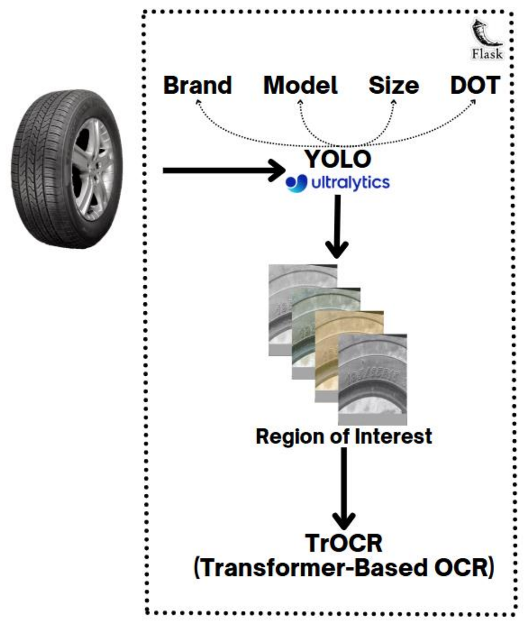


Figure 1. System Architecture for Tyre2Text

To support the services of the *Tyre2Text – Stock Taking* mobile apps, several *resources* are needed so that each *service* can work together and provide optimal results. In the development of *mobile apps* and *the backend* used, Git is chosen as the *version control system*, the code line for *mobile apps* and *the backend* is stored on Github for ease of code line management and collaboration. Using Github Action, the *deployment* process can be done more easily for *backend services*, the process of *building* a line of code into an *artifact* and then *deploying the artifact* into *Azure Container Registry* can be done in a short step. *Azure Container Instance* is used as a container to run *artifacts* that have been *built* into *images* and have been successfully uploaded to *Azure Container Registry,* when the update process is carried out to the code line, then the new images will be automatically updated in *the Azure Container Registry*, and then *Azure Container Instances* will run using the latest line of code that is already stored as *images* in *Azure Container Registry*. *Backend services* are built using Flask, where Flask is the *framework* of the Python programming language.

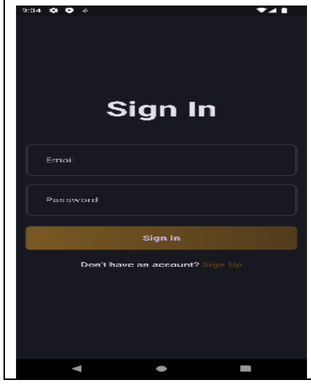
The mobile apps *code line* is built using Flutter, where Flutter is a framework from the Dart programming language, the mobile apps code line is also stored on Github for easy code line management and ease of collaboration. For data storage needs, Supabase is chosen for ease of development, Supabase is a SaaS with a Postgres database, while to accommodate the need for easy config changes, Firebase remote config is used as the tool chosen because of the ease of implementation and ease of configuration. The need for integration between *backend services* and AI classification models and OCR requires special handling, with a large model file size, namely YOLOv8 has a file size of 21.5mb while Transformer-Based OCR or TrOCR has a file size of 240mb, so *Azure Blob Storage* is used as a model storage container. YOLOv8 and TrOCR models that have been *fine-tuned* will be uploaded to *Azure Blob Storage* for ease of management, and when *the backend service* needs access to the model, the *backend service* will easily use both models through *Azure Blob Storage*. This step is also taken to accommodate the file size limit on Github, where currently the maximum size that can be uploaded is 100mb per file (https://docs.github.com/en/repositories/working-with-files/managing-large-files/about-large-files-on-github). By separating the model from the code line and leveraging *Azure Blob Storage*, it also speeds up the *deployment* process from Github into *the Azure Contaienr Registry*, so this step is used in this application.

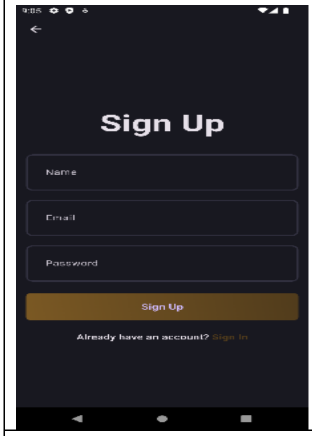
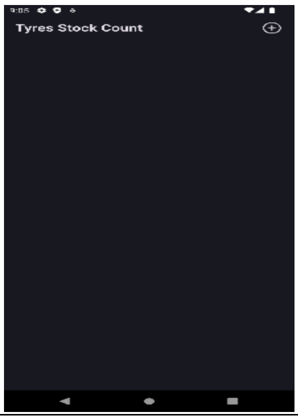
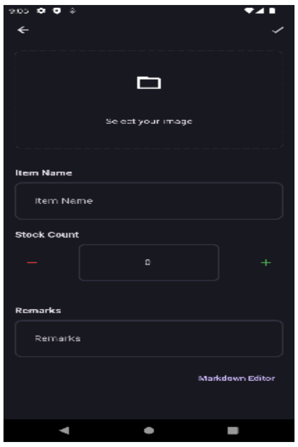
The model used to classify is YOLOv8, the model goes through a fine tuning process with a tire sidewall (https://universe.roboflow.com/tyj-euwyt/tyre-sidewall-text-detection-v2) dataset, the output *classes* identified by this model are Brand, Model, DOT, and Size. And to extract text from images, TrOCR is used, the ROI results issued by the classification process that has been carried out by YOLOv8 will be reprocessed by TrOCR, the TrOCR model used also needs to be done with a fine tuning process, the fine tuning process is carried out with a curved text dataset (SCUT-CTW1500) with the model name used is *microsoft/trocr-small-printed*.

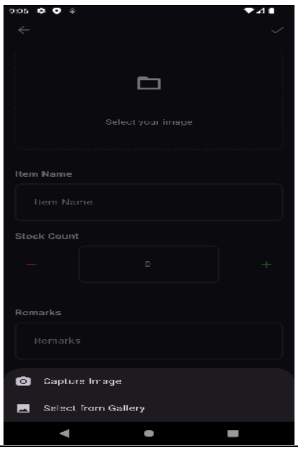
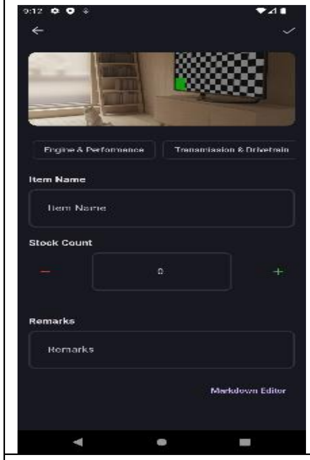
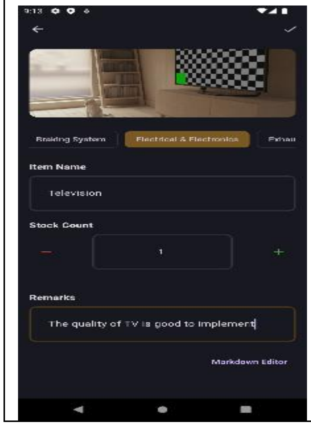
 Flask Brand Model Size DOT YOLO ultralytics Region of Interest TrOCR (Transformer-Based OCR)	Here's an overview of how the flow of a *tyre sidewall* image can be classified and then extracted based on the ROI generated. . As the first step the image will be analyzed and classified by YOLOv8, YOLOv8 will classify the object into Brand, Model, Size, and DOT classes, each detected class will be used as an ROI (Region of Interest) images, the ROI images will then be analyzed by TrOCR to then carry out the OCR process on the detected text, the final output is to return the detected text from each class that has been successfully identified based on ROI.

2. Prototype System Tyre2Text

Then, we continue to create prototype system that can be explained below:

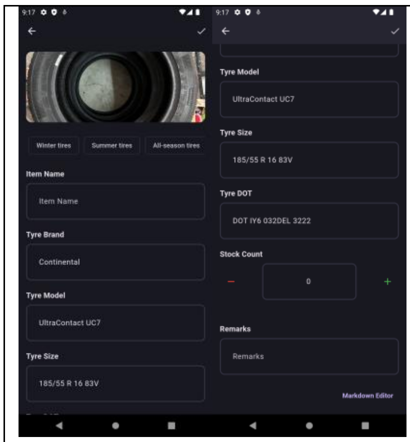
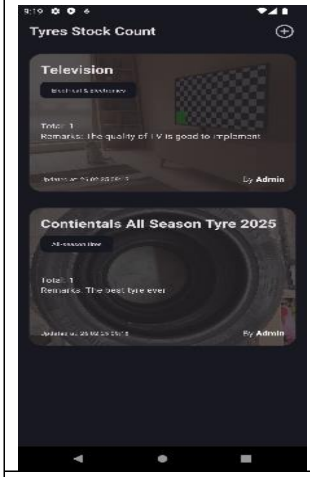
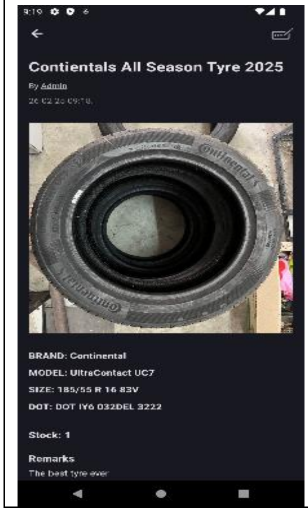
 Sign In Email Password Sign In Don't have an account? Sign Up	To be able to use the Tyre2Text – Stock Opname application, a login process is required first, where only registered users can use this application. This is done to maintain data integrity, only those who have access to the application can manipulate data or analyze data that has been stored. In the login process in this application, a registered and valid email and password are required for the authorization process, if the email and password do not match, an error message will be returned on the display, and if the email and password entered are correct, the user can enter the home page and use this application.

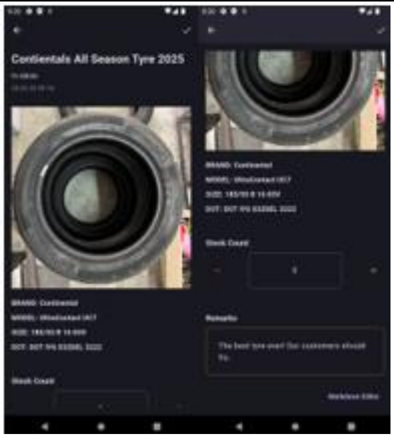
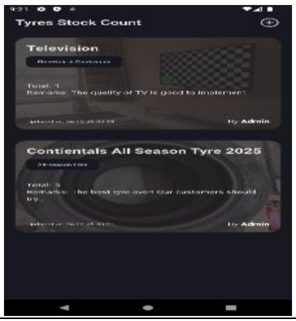
 Sign Up Name Email Password Sign Up Already have an account? Sign In	For users who do not have an email and password to use this application, then the application user can also register, where the registration process requires a valid email and a password that is easy to remember. In addition, users also need to enter a name that will be used as an identifier on the application interface.These three elements, namely email, password and name are needed to support the running of the Tyre2Text – Stock Opname application. After the registration process is complete, users can use the data that has been saved on the sign in page
	After the user has successfully logged in, the user's initial display is the home page. If there is already data stored, the user's home page will contain information that has been stored in the database, such as photos, stock amounts, remarks, the name of the user who input the data, and the date the data was input, but if there is no data stored, an empty home page will appear. On this page there is also a button that can be used to add data, the button is located in the upper right corner. If the button is clicked, the page will move from the home page to the add product page
 Select your image Item Name Item Name Stock Count 0 Remarks Remarks Markdown Editor	To be able to record products along with other information and also record *stock*, users are required to fill in some of the information needed on the *add product page*. As well as product photos, item names, *remarks* or notes, along with the amount of stock to be stored. After the product photo is uploaded to the application, the application will send the photo file to the *backend*, where the *backend* will process the uploaded photo file for the *classification* process along with *the text detection* process by utilizing *artificial intelligence*. In addition to uploading product photos, users are also required to fill in the name information of the item to be saved, and then users can also add notes related to the product to be saved. To store stock data, users can use the '-' and '+' buttons, but alternatively, users can also directly fill in the number of products to be stored in the *textbox* that has been provided. After the data input process is complete, users can save the data that has been filled in the database by pressing the *checkmark button* in the upper right corner.

 9:05 Select your image Item Name Item Name Stock Count 0 Remarks Remarks Capture Image Select from Gallery	On the previous page, it was explained how to add product information along with the amount of stock to be stored, one of the required product information is a product photo. To be able to upload product photos into the Tyre2Text – Stock Photography application, it can be done in two ways, including by directly using the camera feature to capture images or by using the feature to select images from the photo gallery available on each *device* that is being used. The two methods have no difference in carrying out the stock recording process, the difference is only in the part of how users can upload photos to be saved
 9:12 Engine & Performance Transmission & Drivetrain Item Name Item Name Stock Count 0 Remarks Remarks Markdown Editor	If the user uploads an image that cannot be classified by the AI model that has been used, the application will display a simple display containing the display of the photo that has been successfully uploaded, then a *text box* to fill in the name of the item and *remarks*, as well as *a text box* and a button to be able to fill in the amount of stock to be saved. Images that can be classified with AI models that are currently used are limited to images that show the side of *tires* or tires, apart from these images, it is likely that the process of classification and text extraction from images that have already been uploaded cannot be carried out.
 9:13 Braking System Electrical & Electronics Exhau Item Name Stock Count Remarks Markdown Editor Television 1 The quality of TV is good to implement	On the *add product* page, after the user has successfully uploaded a photo of the product that has been cooled, the user will see the display of the image that has been selected, then the user will also be required to fill in some other information needed for the recording process. The information needed includes the product category to be stored, whether the electric system or braking system and other categories. In addition, users are also expected to fill in the name of the item to be saved, *remarks* or notes related to the item to be saved, along with the amount of stock that will be stored in the database. After the process of filling in the information is complete, the user can press the *checkmark* button in the upper right corner to save the information in the database.

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

 9:17 Winter tires Summer tires All-season tires Item Name Item Name Tyre Brand Continental Tyre Model UltraContact UC7 Tyre Size 185/55 R 16 83V 9:17 Tyre Model UltraContact UC7 Tyre Size 185/55 R 16 83V Tyre DOT DOT IY6 032DEL 3222 Stock Count − 0 + Remarks Remarks Markdown Editor	On the *add product page,* if the uploaded image is in the form of an image showing the side of the tire, the AI model will be able to perform the classification process and the text extraction process from the image that has been successfully uploaded. The significant difference between images that are successfully classified and extracted and those that are not successfully is the number of *textboxes* displayed on the app's interface display. For images that are successfully classified as tire images, there will be additional *textboxes*, including *brand, model, size,* and DOT textboxes, but if it doesn't work, a simple display will appear that has been described earlier. *The textbox brand, model, size,* and *DOT* will be automatically filled in based on the results of text extraction of the classified images, so users only need to fill in information related to the name of the product to be saved, *remarks* or notes related to the product to be saved, and the amount of stock to be stored in the database. However, this page also does not limit application users to justify the results of text extraction that have been successfully carried out by the AI model, users can also add or subtract words that will be stored in the *textbox brand, model, size,* and DOT to match what the user wants
 9:19 Tyres Stock Count Television Electrical & electronics Total: 1 Remarks: The quality of TV is good to implement Updated at [illegible] By Admin Contientals All Season Tyre 2025 All-season tire Total: 1 Remarks: The best tyre ever Updated at: 26 02 25 09:18 By Admin	After the user completes the product information along with the number of stocks and then saves the information into the database by pressing the *checkmark* button in the upper right corner, the user display will be directed to the home page. On the home page, users will see information that has been successfully saved in the database, the information display will be grouped according to the product information that has been saved, such as the name of the product, the number of stock, *remarks* or notes, as well as the name of the user who saved the data and the date when the data was saved. In addition, product photos that have been successfully uploaded will also be displayed as *a background* display to facilitate the analysis process.
 9:19 Contientals All Season Tyre 2025 By Admin 26 02 25 09:18 BRAND: Continental MODEL: UltraContact UC7 SIZE: 185/55 R 16 83V DOT: DOT IY6 032DEL 3222 Stock: 1 Remarks The best tyre ever	If the user presses the product section on the home page, the user will be directed to the product detail page, on which the user can see the details of the information that has been successfully stored in the database. If the user feels that the information that has been stored is not accurate, such as the user wants to change the amount of stock or just add a note in the *remarks* section, then the user can press *the edit icon* in the upper right corner. However, other information such as the product name, and even *the brand, model, size*, and DOT cannot be changed to maintain data integrity, if there is a discrepancy for the information then the user is expected to store the new product with the adjusted data

	If the user does the *product editing process,* the user will be directed to a different page, where on that page the user can only make changes to stock information and *remarks* or notes. On this page, users can add or even decrease the number of stock that has been successfully saved if there is a mistake before, users can also add a note in the remarks section to just provide a reason why the product editing process is carried out. If the user has made adjustments to the information to be saved, the user can save the changes by pressing the *checkmark* button in the upper right corner on the *product edit page.*
 Tyres Stock Count Television Total: 1 Remarks: The quality of TV is good to implement By Admin Contientals All Season Tyre 2025 Total: 5 Remarks: The best tyre ever! Our customers should try... By Admin	After the product editing process is carried out and the user saves the change information in the database, the user will be directed back to the home page with an updated information display based on the data that has been successfully stored in the database.

CONCLUSION

This research produces an android application integrated with API, where the API acts as a service that performs image processing. Images from the stock opname application uploaded by users will be classified into four classes, including Brand, Model, Size, and DOT. The results of the classification process will then be reprocessed in pre-processing so that the level of accuracy when extracting text increases, the results of pre-processing will then be used for text extraction. To increase the accuracy of the text that will be obtained, post-processing will be carried out by validating the LLM. The results of the processing that has been done will be sent by the API to the android application, to then be displayed to the user. Thus, users only need to take a photo of the side view of the tire, and information about the tire can be obtained, users can also make adjustments to the tire information if they feel it is not appropriate. Users will then be able to input the number of stock items and enter notes related to the items being input, then can save the information. The information that has been saved can also be changed if adjustments need to be made another day. It is hoped that in this application there will be more types of goods that can be classified so that the stock recording process can be made easier.

AUTHOR CONTRIBUTIONS

Ahmad Nurul Fajar: Conceptualization, Literature Review, Supervision, Validation, Writing. Yoeka Virya Adhitthana: Conceptualization, Methodology, Model Training, Application Development, Writing. Veronica Lestari Jauw: Supervision, Validation. Nunung Nurul Qomariyah: Supervision, Validation. All authors discussed and contributed to the final manuscript.

OPEN DATA AVAILABILITY STATEMENT

The data used in this research are already public and can be accessed at https://universe.roboflow.com/hibahproject/tyre-to-text.

ACKNOWLEDGMENT

This work was financially supported by the collaborationpartners, Continental Tyre Technology Centre Malaysia and Bina Nusantara (BiNus) University. The funding from BiNus comes from BINUS International Research – Applied

program, titled "Enhancing Road Safety through Image-to-Text Tire Information Application System." The contract number is 097/VRRTT/VII/2024, dated July 2, 2024.

REFRENCES

[1] H. Wang, C. Pan, X. Guo, C. Ji, and K. Deng, "From object detection to text detection and recognition: A brief evolution history of optical character recognition," WIREs Computational Statistics, vol. 13, no. 5, Jan. 2021. doi:10.1002/wics.1547
[2] N. Chicu, A.-L. Prioteasa, and A. Deaconu, "Current Trends and Perspectives in Tyre Industry," Studia Universitatis „Vasile Goldis" Arad – Economics Series, vol. 30, no. 2, pp. 36–56, Jun. 2020, doi: https://doi.org/10.2478/sues-2020-0011
[3] M. A. D. Wickrama, D. S. C. Dharmakeerthi, S. A. Balasooriya, U. M. I.S. Ekanayake, M. P. Gamage, and N. Amarasena, "Mobile Based Solution for Vehicle Assistance," IEEE Xplore, Dec. 01, 2021. https://ieeexplore.ieee.org/abstract/document/9671196.
[4] P. Mohan, A. Pahinkar, A. Karajgi, L. D. Kumar, R. Kasera, A. K.Gupta, and S. J. Narayanan, "Multi-contrast convolution neural network and fast feature embedding for multi-class tyre defect detection," in 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), Nov 2020, pp. 1397–1405.
[5] W. Kazmi, I. Nabney, G. Vogiatzis, P. Rose, and A. Codd, "An efficient industrial system for vehicle tyre (tire) detection and text recognition using deep learning," IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 2, pp. 1264–1275, 2021.
[6] M. J. Kalsher, M. S. Wogalter, K. R. Laughery, and R. W. Lim, "Consumer Knowledge of Tire Maintenance and Aging Hazard," Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 49, no. 18, pp. 1757–1757, Sep. 2005, doi: https://doi.org/10.1177/154193120504901817.
[7]]. N. Chicu, A.-L. Prioteasa, and A. Deaconu, "Current Trends and Perspectives in Tyre Industry," Studia Universitatis „Vasile Goldis" Arad – Economics Series, vol. 30, no. 2, pp. 36–56, Jun. 2020, doi: https://doi.org/10.2478/sues-2020-0011
[8] V. Ozdemir, O. Urhan, M. Ozgen, A. Calı, skan, A. H. Ozcan, R. Elioz, and H. Kara, "Tyre (tire) brand and size detection with computer vision," in 2022 30th Signal Processing and Communications Applications Conference (SIU), 2022, pp. 1–4.
[9] Tomborski.M, Rojek.I., Mikolajewski.D. " Revolutionizing Tire Quality Control: AI's Impact on Research, Development, and Real-Life Applications, *Appl. Sci.* 2023, *13*(14), 8406; **https://doi.org/10.3390/app13148406**
[10] https://dl.acm.org/doi/10.1145/1028664.1028736. Mobile-D: an agile approach for mobile application development
[11] Parkpoom Lertsawatwicha, Phumidon Phathong, Napatsorn Tantasanee, Kotchakorn Sarawutthinun Thitirat Siriborvornratanakul, "A novel stock counting system for detecting lot numbers using Tesseract OCR. Int. j. inf. tecnol. (January 2023) 15(1):393–398. https://doi.org/10.1007/s41870-022-01107-4
[12]Adamos Daios , Alexandros Xanthopoulos,Dimitrios Folinas , Ioannis Kostavelis. "Towards automating stocktaking in warehouses: Challenges, trends, and reliable approaches". Procedia Computer Science, Volume 232, 2024, Pages 1437-1445. https://doi.org/10.1016/j.procs.2024.01.142