**PTA-FTSM-2025-A191848**

# MACHINE VISION-BASED PARKING SPACE DETECTION FOR EFFICIENT PARKING MANAGEMENT

TIAN KUO, Elankovan A. Sundararajan

Faculty of Information Science & Technology
Universiti Kebangsaan Malaysia
43600 Bangi, Selangor



## ABSTRACT

This project presents a real-time parking space monitoring system based on machine vision, developed to address parking challenges at urban campuses such as the Faculty of Information Science & Technology (FTSM), UKM. It solves the problem of users being unaware of parking availability by using a fixed camera and the YOLOv8 deep learning model to detect parked vehicles from periodic screenshots. The number of available spots is calculated and updated in real time to Firebase, which feeds data to both a mobile app (developed with Jetpack Compose) and a web interface (built with HTML5/JavaScript). An incremental, modular development strategy was adopted, including image processing, local SQLite storage, cloud synchronization, and user interfaces. The system supports both CPU and GPU inference, making it suitable for resource-limited environments. Results demonstrate high detection accuracy and smooth user access to live data, improving user experience, reducing unnecessary vehicle movement, and contributing to smart campus infrastructure. The architecture is scalable and adaptable for expansion to other parking lots or institutions.

# INTRODUCTION

Urbanization and the increasing number of vehicles have led to significant challenges in traffic management, especially in infrastructure-limited areas like university campuses. One major contributor to urban congestion is the inefficiency in locating parking spaces. According to Tanenbaum & Van Steen (2017), in complex distributed systems such as urban environments, inefficient resource discovery can result in cascading delays, congestion, and reduced system throughput. This insight applies directly to parking, where the lack of real-time availability information forces drivers to search manually, contributing to 30~40% of total city traffic congestion.

Moreover, the underutilization of existing parking spaces further aggravates the issue. Many lots have vacant spots that remain unused simply because drivers are unaware of them. Without dynamic guidance, large facilities often fail to operate efficiently. A report by AutoNavi highlighted that in cities like Shenzhen, the "parking difficulty index" reaches 11.6 indicating that approximately 12 out of every 100 navigation attempts fail due to unavailable parking, causing delays and forcing drivers to re-navigate (Xinhua News Agency, 2022).

This mismatch between parking supply and accessibility underscores the need for intelligent systems that provide real-time monitoring and feedback. Traditional methods such as static signage or manual attendants are no longer adequate in fast-changing urban settings. Smart parking systems based on machine vision offer a scalable solution to detect, compute, and deliver parking availability information to users in real time, thereby enhancing resource utilization and alleviating traffic congestion.

Despite some improvements in infrastructure, the lack of real-time data integration continues to limit the effectiveness of parking management especially in high-traffic areas like university campuses.

To address campus parking challenges, this project develops a machine vision-based system that provides real-time parking availability. It captures periodic screenshots from a fixed camera, processes them using the YOLOv8 algorithm to detect vehicle occupancy within predefined parking zones, and determines space availability through spatial comparison.

The results are uploaded to Firebase Realtime Database and accessed via a cross-platform front end: a web dashboard and a Jetpack Compose based Android app, both offering intuitive map-based interfaces. Cloud synchronization ensures consistent, real-time updates across platforms.

By integrating machine vision, cloud data, and modern UI technologies, this scalable system enhances parking efficiency and supports smart infrastructure goals in campuses and urban environments.

# LITERATURE REVIEW

The development of parking space monitoring systems has progressed significantly, evolving from traditional sensor-based approaches to intelligent camera-driven machine vision systems. These methods have been designed to address key challenges such as detection accuracy, scalability, implementation cost, and environmental adaptability. To provide a solid technical foundation for this project, it is important to review previous studies and assess the trajectory of these technologies.

In the early stages of intelligent parking management, sensor-based systems dominated. These systems employed different types of sensors to detect vehicle presence, each with unique strengths and limitations. Ultrasonic sensors, typically mounted above or beside parking spots, measure sound wave reflections to detect vehicles. They offer high precision but require installation in every individual space, which increases hardware cost and complexity. Infrared sensors, both active and passive, detect vehicle presence through reflected light or thermal radiation. While cost-effective and widely used, they are sensitive to environmental factors such as lighting and weather. Geomagnetic sensors, embedded beneath the surface, detect changes in magnetic fields caused by nearby vehicles. These are energy-efficient and reliable but require ground modification during installation, leading to higher maintenance costs.

Machine vision was initially introduced for license plate recognition but has expanded to include image-based parking occupancy detection. Research has shown that deep learning models, particularly the YOLO series, can achieve detection rates exceeding 95% under controlled conditions. Smith et al. and Johnson & Lee (2020) reported over 92% detection accuracy across different lighting and weather conditions (Smith, J., et al. 2021; Johnson, R., & Lee, S. 2020). Additionally, GPU-accelerated pipelines support latencies below 2 seconds, enabling real-time decision-making in busy scenarios (Brown, L., et al. 2021). Vision-based systems also incorporate image enhancement and normalization techniques to maintain accuracy above 85% in low-light or adverse weather (Zhang, Y., & Wang, H. 2019). Unlike sensor systems, machine vision can monitor multiple spaces with fewer hardware units. The National

Parking Association reports successful implementations across diverse urban settings (National Parking Association, 2022; Kim, T., & Roberts, M. 2020).

Sensor-based systems offer high point-specific accuracy and straightforward operational principles and can adapt to a range of environments through different sensor types. However, they require installation in every parking space and are more vulnerable to environmental interference. In contrast, machine vision-based systems reduce hardware cost by allowing a single camera to monitor multiple spaces, simplify deployment and expansion, and provide superior real-time performance. However, they may face reduced accuracy under extreme lighting conditions and require greater computational power.

Recent advancements in deep learning and YOLO based object detection have further strengthened the capabilities of machine vision systems. These models exhibit resilience to partial occlusions and lighting variations, making them ideal for real-world parking environments. To increase spatial coverage and minimize blind spots, systems are increasingly adopting multi-camera fusion and panoramic image stitching. These approaches combine spatial data from multiple angles, enhancing robustness and reducing single-point failures.

Cloud integration has also become a key component of modern parking systems. Real-time cloud platforms such as Firebase enable cross-device synchronization, allowing users to access live parking availability through both mobile apps and web interfaces.

In conclusion, parking space monitoring technologies have evolved considerably, with machine vision becoming a central component in modern smart parking systems. While sensor-based solutions offer mature technology and precision, they are limited by high deployment and maintenance costs. In contrast, machine vision approaches particularly those using deep learning models like YOLO offer enhanced scalability, flexibility, and real world accuracy. The integration of multi-camera input, real-time processing, and cloud-based synchronization lays the foundation for the intelligent and reliable parking solutions proposed in this project.

# RESEARCH METHODOLOGY

## USER NEEDS AND FUNCTIONAL EXPECTATIONS

To meet real-world needs, the system was designed based on campus parking behavior and stakeholder expectations. It provides real-time parking availability using static screenshots processed by a YOLOv8 model, with data synchronized via Firebase for web and Android access. This reduces time spent searching for parking, lowers fuel consumption, and enhances user satisfaction. The interface is simple, responsive, and cross-platform compatible.

Functionally, the system detects parking occupancy in real time, displays current space availability, stores historical data in SQLite, and syncs live updates through Firebase. It works across devices and browsers, ensuring consistent user experience.

Non-functional requirements include image processing every 30 seconds, system responsiveness under 5 seconds, 90% uptime, and detection accuracy above 70%

in difficult conditions. The UI is designed for ease of use, achieving high user satisfaction, and ensuring secure data communication and storage.

Development uses a high-performance PC, YOLOv8 with Python, and a weatherproof surveillance camera. The deployed system relies on standard hardware and software, including a backend with Firebase and SQLite, and a front-end app using Jetpack Compose and web technologies.

Communication is handled through secure HTTPS over TCP/IP with Firebase. Local devices connect via Ethernet or Wi-Fi, and image data is collected using proprietary client software.

These requirements ensure the system is robust, scalable, and ready for practical deployment in real-world environments.

**SYSTEM ARCHITECTURE**

The system adopts a client–server architecture, which is widely used in real-time monitoring applications requiring centralized computation and synchronized data delivery. The backend server handles core tasks such as image preprocessing, vehicle detection using YOLOv8, and parking space availability analysis, while the frontend (Web and Android App) serves as a user interface for accessing real-time data.

This architectural model provides several key advantages. It ensures system scalability, allowing for the easy addition of multiple camera inputs and seamless expansion to new parking lots with minimal structural adjustments. In terms of maintainability, backend updates can be performed independently of the frontend, reducing the need for frequent client-side changes. The architecture also enhances performance by offloading compute-intensive tasks—such as detection and analysis—to the server, ensuring that client applications remain lightweight and responsive. Furthermore, cross-platform access is enabled through consistent real-time data delivery to both web browsers and Android applications.

By clearly separating responsibilities between components, the system achieves improved performance, reliability, and flexibility—qualities essential for intelligent parking management on an urban scale (Tanenbaum & Van Steen, 2017; Kurose & Ross, 2021; Goodfellow et al., 2016).

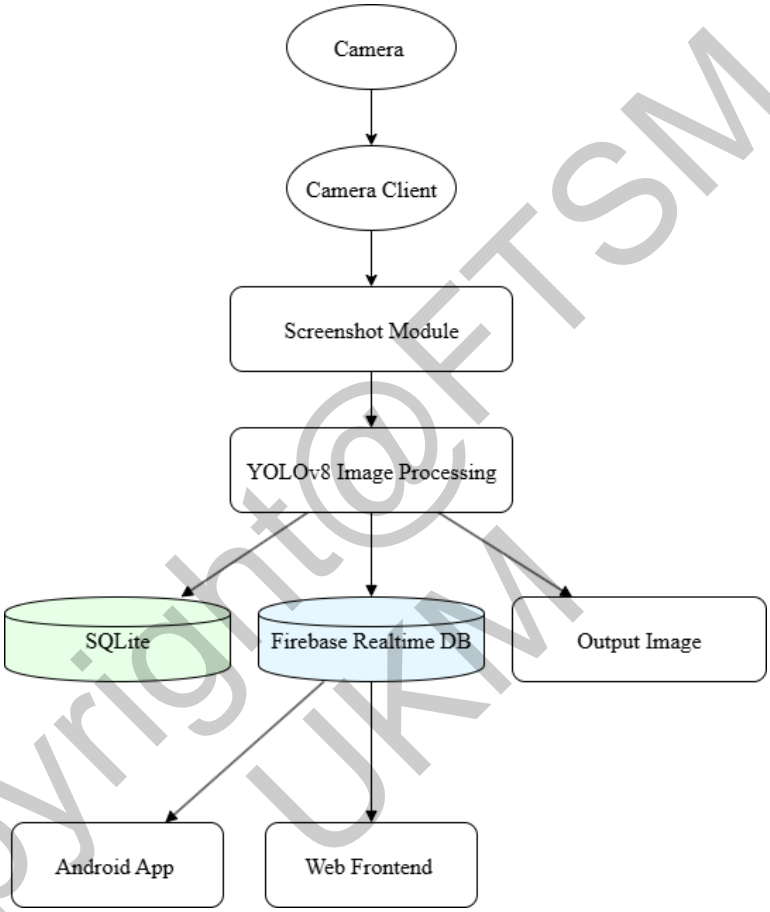


Figure 3.1 System Architecture

Figure 3.5 illustrates the system's architecture and module interactions.

This system integrates several modules working together to detect vehicles and monitor parking availability in real time. It starts with periodic screenshots of the parking lot captured via the camera client and pyautogui. These images are preprocessed and analyzed by the YOLOv8 model to detect vehicles within a defined area. The detected positions are compared with known parking spots to identify which spaces are free.

The latest parking data is uploaded to Firebase Realtime Database for real-time access by web and Android clients, while historical data is stored locally in SQLite for analysis.

The front-end provides users with interactive maps displaying parking status using Google Maps API.

Although modular and logically separate, these components are integrated through local function calls and real-time synchronization, ensuring efficient, scalable, and reliable operation.

**INTERFACE DESIGN**

The interface of the parking monitoring system is designed with a strong focus on visual clarity, intuitive layout, and real-time responsiveness. It supports both web and Android platforms, ensuring consistent user experience across devices. Given that most users access the system while on the move, the design prioritizes simplicity and efficiency—allowing users to access essential parking availability information with minimal interaction (Zhang & Wang, 2021).

The system relies on Firebase Realtime Database for real-time data updates. Parking lots are displayed on a map using the Google Maps API, with color-coded markers indicating availability: green for ample spaces, orange for limited, and red for near full. Tapping a marker opens a detailed view showing the layout and real-time status of each space—green for free, red for occupied—helping users instantly assess remaining availability and spot locations.

# RESULT

## SYSTEM DEVELOPMENT

The parking monitoring system is developed using Python for backend processing and HTML/CSS/JavaScript for frontend visualization. At its core, the system integrates real-time image acquisition, deep learning-based vehicle detection, and multi-platform data presentation. Surveillance cameras, fixed in position, streams live footage to a local client application. The backend leverages PyAutoGUI to periodically capture screenshots from this video feed. After contrast enhancement, the images are processed by the YOLOv8x model to detect vehicle positions.

Parking space occupancy is determined by comparing the detection results against pre-annotated central coordinates of each slot. This method allows for reliable, real-time status updates. The detection outcomes are written to Firebase Realtime Database for instant frontend access and simultaneously logged in a local SQLite database to preserve historical records and support future analysis.

The frontend, including both a web app and an Android app, retrieves data from Firebase and presents parking availability and timestamps via a user-friendly graphical interface. This ensures synchronized, responsive updates across platforms.

This architecture, as illustrated in Figure 4.1 (Development Flow Chart) and Figure 4.2 (Workflow Diagram), demonstrates a modular and scalable design. The dual-layer data storage enhances system reliability and accessibility, while the use of YOLOv8x ensures robust detection performance. Together, these components support the system's real-world deployment in campus environments and allow for future expansion into broader urban applications.

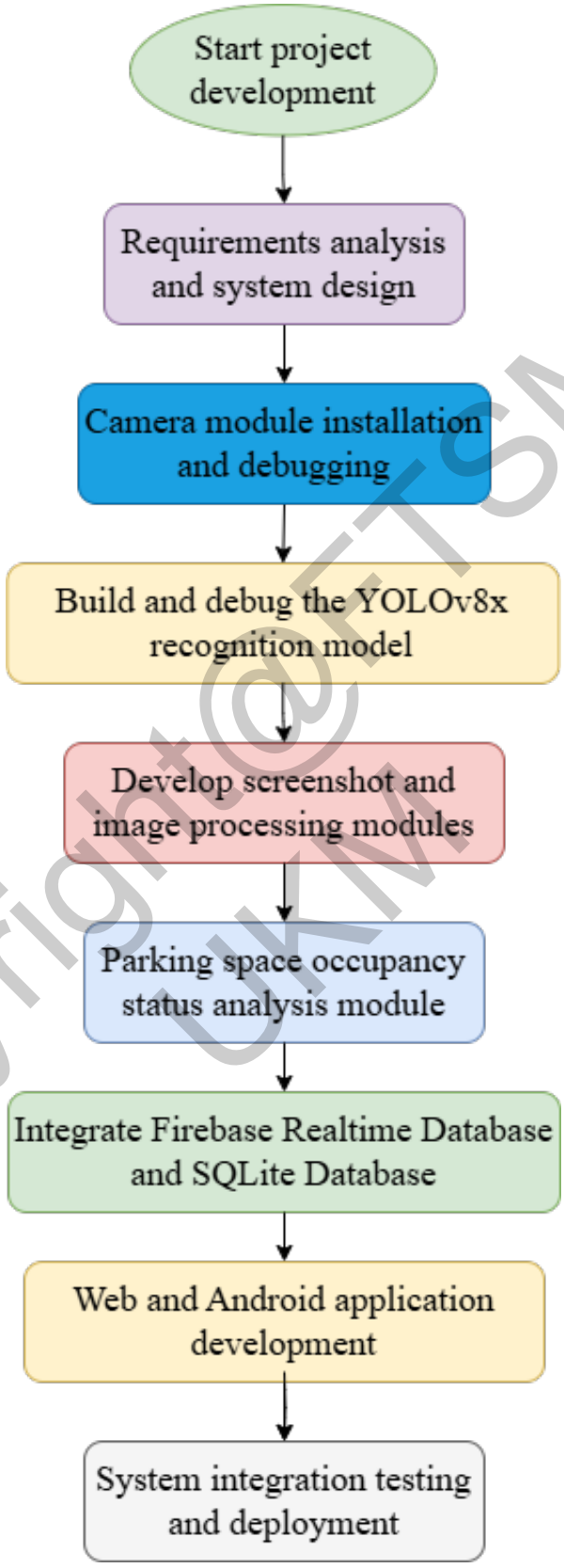


Figure 4.1 Development Flow

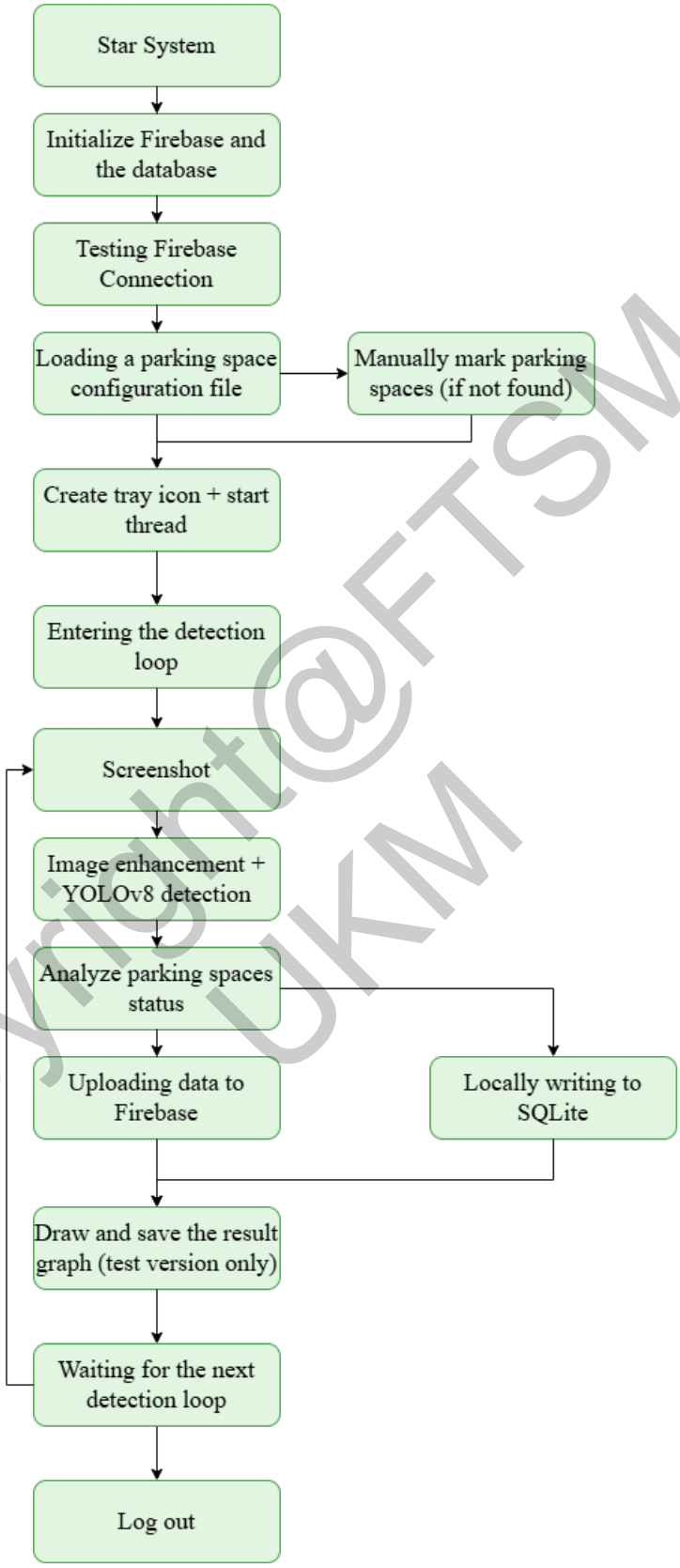


Figure 4.2 Workflow Diagram

## INTERFACE OVERVIEW

During the system development phase, the interface design was implemented as planned and optimized according to actual usability requirements. The Web interface was built using HTML/CSS/JavaScript and Google Maps API, where users can view parking locations, click on map markers, and access detailed status pages; while the Android application was implemented using Jetpack Compose, which maintains the same interaction logic while being more adaptable to mobile screens and touch operations.

The interfaces of both platforms obtain parking data through the Firebase real-time database to ensure that the displayed information is up to date. The system also uses dynamic colour coding to enhance the visualization of remaining parking space information. Through iterative development and user feedback, the interface finally achieved the goal of being lightweight, intuitive, and effectively supporting real-time parking decisions.

The following is a screenshot of the interface after the system is developed:

### Web

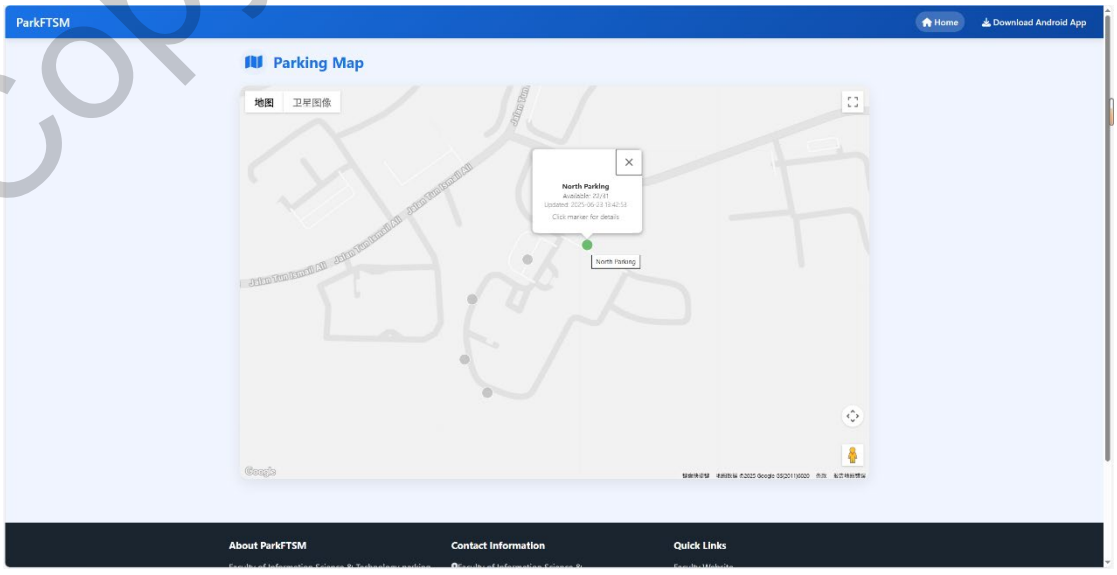


Figure 4.3 Dashboard of Web

**Dashboard**: This page uses Google Maps to display all parking areas of FTSM, with color coding to indicate availability. Click to jump to the details page.

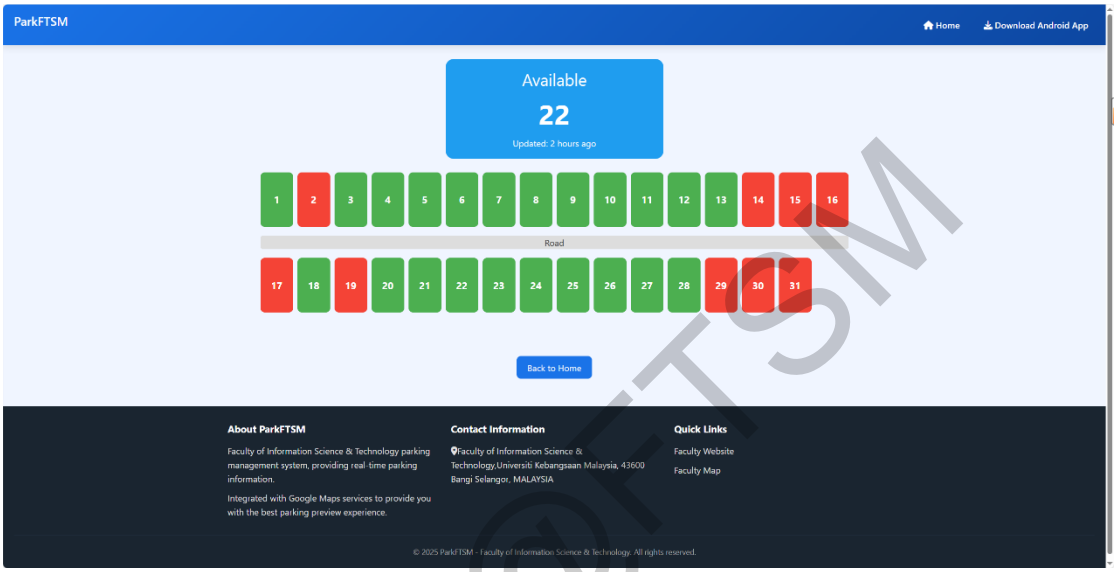


Figure 4.4 Detail Page of Web

**Details page**: Displays the real-time occupancy status of the selected parking lot, with color markings to distinguish between available and occupied spaces.

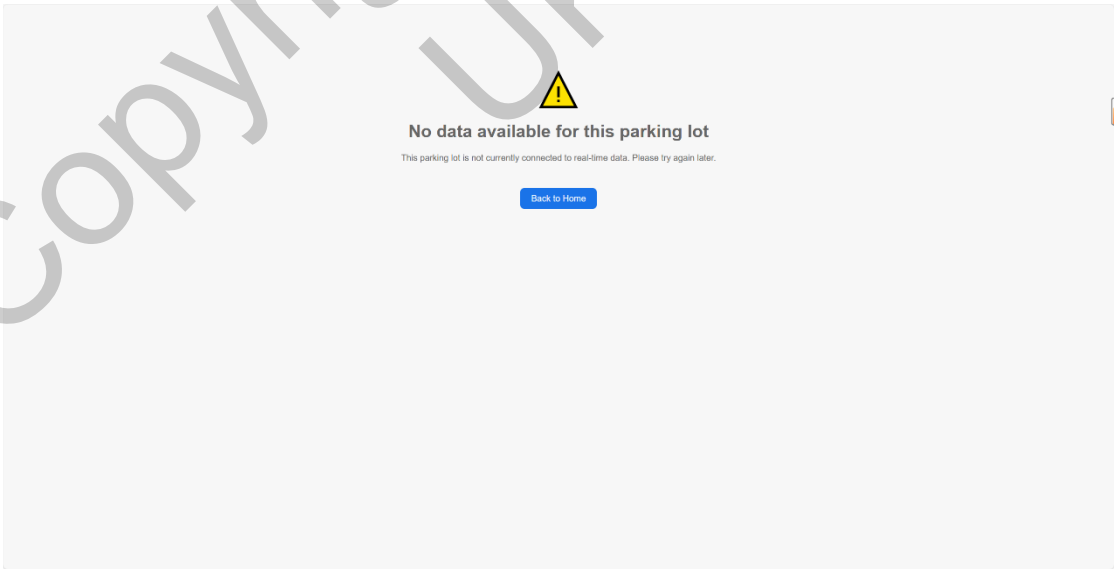


Figure 4.5 No Data Page of Web

**No data page**: This page is displayed when there is no data for the selected parking lot.

**Android Application**

The following figure 4.12 shows each interface of the Android application**:**

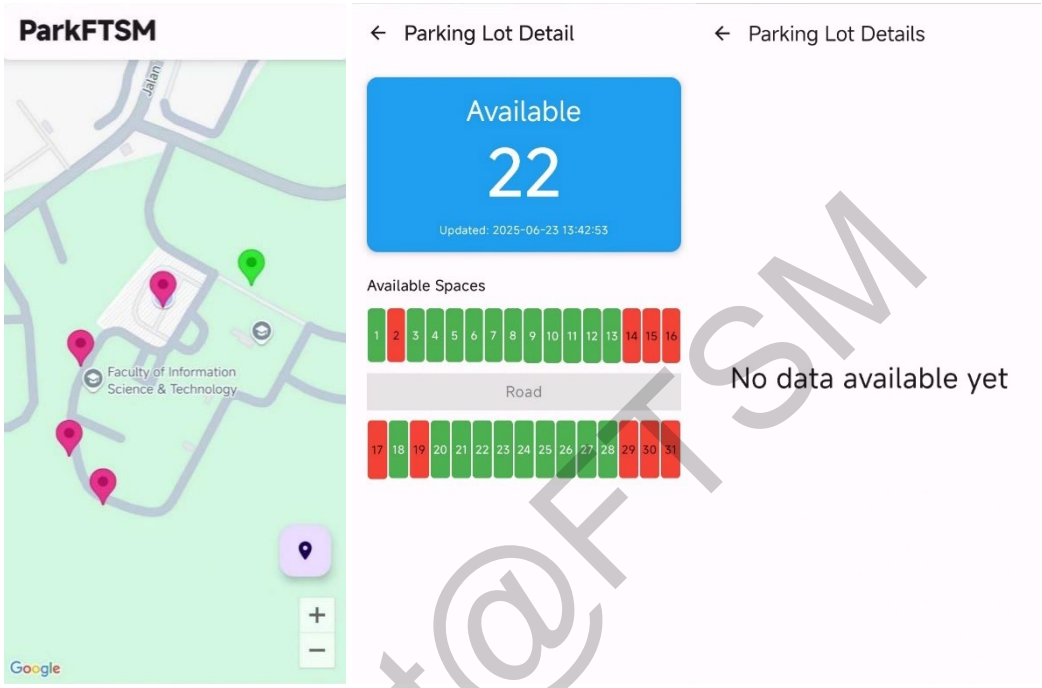


Figure 4.6 Android Application Interfaces

**Home page** : Similar to the web version, parking spot markers are displayed via Google Maps SDK.

**Details page** : Click a parking area marker to jump to the details page, which displays the parking space layout and parking space status of the selected parking lot.

**No data page**: This page is displayed when there is no data for the selected parking lot.

**Backend interface**

The backend interface of the system operates as a lightweight background program integrated into the system tray. It is designed to launch automatically upon system startup and continuously perform periodic screenshot capture and vehicle detection without requiring user intervention. The system tray icon also provides basic control functions: users can right-click the icon to pause or resume the monitoring process, or to safely exit the program when needed.

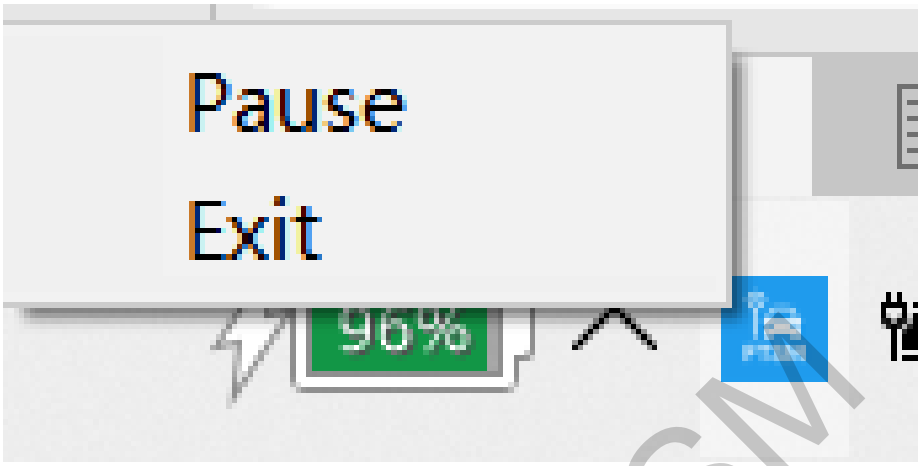


Figure 4. Interface of System Tray

In conclusion, the user interface of the parking monitoring system was designed with a focus on simplicity, real-time responsiveness, and platform adaptability. From the initial layout concepts proposed in the project planning stage to the final implementation across web, Android, and backend tray applications, the UI effectively supports intuitive user interaction and operational transparency. By integrating Google Maps visualization, real-time Firebase data retrieval, and lightweight control mechanisms, the interface component contributes significantly to the overall usability and system effectiveness.

**TESTING**

System testing was carried out to ensure the reliability, accuracy, and usability of the machine vision-based parking monitoring system. The testing phase included three major components: white-box testing, user acceptance testing (UAT), and usability testing.

White-box testing focused on validating the core backend logic, particularly the vehicle detection and parking space analysis modules. These tests verified whether detected bounding boxes matched the predefined parking slot coordinates using image-based inputs. Two representative test cases were executed, and both passed successfully with accurate classification of occupied and unoccupied spaces. The summarized white-box test results are shown in Table 4.1, indicating consistent logic execution and correct Firebase updates.

Table 4.1 Summary of White Box Testing Result

| Test Case ID | Test Item | Conclusion | Description |
|---|---|---|---|
| Test Case 1 | Vehicle Recognition Function (detect_vehicles ()) | Passed | All vehicles correctly identified with inference time consistently between 0.3-0.6 seconds. Misidentification rate below 5%. |
| Test Case 2 | Parking Space Occupancy Recognition (analyze_parking ()) | Passed | System accurately identifies whether parking spaces are occupied by vehicles.Output available_ spot_ids matches manual verification results. |
| Test Case 3 | upload_to+firebase (parking_info: Dict) save_parking_info_local (parking_info : Dict) | Passed | The system can correctly upload the latest data to the Firebase Realtime Database and save it to the SQLite database with a response speed of less than 10 seconds. |



User acceptance testing evaluated whether the system met user expectations for real-time parking information retrieval. Test participants accessed parking data through both the responsive web dashboard and Android mobile application under typical operating conditions. As summarized in Table 4.2, UAT confirmed successful synchronization between frontend and backend systems, with real-time updates displayed correctly. No major functional errors were reported during this testing.

Table 4.2 Summary of UAT Testing Result

| Test Case ID | Test Platform | Conclusion | Description |
|---|---|---|---|
| Test Case 4 | Web | Passed | The page loads quickly, the parking lot marking and jump functions are stable, and the parking space status display is accurately synchronized with Firebase data. |
| Test Case 5 | Android Devices | Passed | Users can smoothly click on the map makers and jump to the details page. The interface responds smoothly, and |

| Test Case ID | Test Platform | Conclusion | Description |
|---|---|---|---|
| | | | the parking status is displayed consistently. There are no crashes or freezes during the test. |

Usability testing was conducted using a questionnaire distributed to 9 students from the Faculty of Information Science and Technology (FTSM), including students, academic staff, and administrative personnel. The questionnaire assessed three dimensions: ease of use, interface clarity, and overall user satisfaction. Results, summarized in Table 4.3, show that 85% of users rated the system as easy to use and intuitive. The average satisfaction score was 4.3 out of 5, exceeding the target threshold of 4.0. These findings confirm that the system is accessible, responsive, and well received by its intended users.

Table 4.3 Summary of Usability Testing Results

| Evaluation Criteria | Average rating |
|---|---|
| Interface usability | 4.89 |
| System interaction experience | 4.67 |
| System responsiveness | 5 |
| Operational simplicity | 4.8 |
| Overall satisfaction | 5 |

Overall, the testing phase demonstrated the robustness of the backend detection algorithm, the seamless performance of Firebase synchronization, and the high usability of the frontend interfaces. The system achieved stable functionality under real-world campus conditions, supporting its deployment as a smart parking solution.

# CONCLUSION

This project successfully developed a machine vision-based smart parking monitoring system aimed at improving the real-time detection and visualization of parking space availability, particularly in a campus setting. The system integrates a YOLOv8-based detection module, which processes periodic screenshots to determine vehicle presence, and synchronizes the analysed results to both Firebase and a local SQLite database. With user interfaces built for both web and Android platforms, the system allows users to access live parking data anytime, with clear visual indicators and fast responsiveness. The completed system was tested extensively and demonstrated strong performance, achieving a vehicle detection accuracy of over 97% and receiving positive feedback from users in usability testing.

Compared to traditional sensor-based solutions, the machine vision approach adopted in this project provides several practical advantages. A single fixed-position camera can monitor multiple parking spaces without the need for per-slot hardware installations, significantly reducing deployment and maintenance costs. In addition, the system is highly scalable, and adaptable updates can be applied through software modifications without the need for invasive infrastructure changes. Its cloud-local hybrid architecture ensures real-time synchronization while preserving local data for historical analysis, making the system especially suitable for institutions like universities and public parking areas that require both live monitoring and long-term data retention.

Despite its strengths, the current version of the system does present certain limitations. The performance may be affected under extreme lighting conditions, such as glare or darkness. The system relies on a fixed camera angle, and occlusions or environmental changes could impact detection accuracy. Additionally, parking space positions must be manually annotated, and the current system is limited to a single parking lot scenario. These constraints, however, are not fundamental, and can be addressed through future enhancements.

Looking ahead, several directions can be pursued to improve the system's functionality and applicability. Transitioning from static image capture to video stream processing would enhance real-time performance and fluidity. Incorporating dynamic camera controls (such as PTZ functionality) could reduce blind spots and expand coverage. Automated parking slot detection using AI-based methods would eliminate manual configuration. Furthermore, the system can be enriched with features such as license plate recognition, parking duration tracking, and violation alerts. Integration with external systems—such as campus access control, navigation apps, or e-payment platforms—would enable the creation of a more comprehensive smart parking ecosystem. These enhancements would make the system more robust, flexible, and aligned with the evolving needs of smart city environments.

## ACKNOWLEDGEMENT

I am deeply grateful to my research supervisor, Associate Professor Ts. Dr. Elankovan A. Sundararajan, for his invaluable guidance, unwavering support, and profound expertise throughout my research journey. His encouragement and constructive feedback have been instrumental in shaping this work.

I would also like to extend my heartfelt appreciation to the Faculty of Information Science and Technology (FTSM) for providing me with the resources and opportunities to pursue my studies. Special thanks to all the lecturers and staff at FTSM for their kindness, mentorship, and dedication to fostering a conducive learning environment.

To my fellow students and friends at FTSM, thank you for your camaraderie, assistance, and for making my time here both enjoyable and memorable. Your support has been a source of motivation throughout this journey.

I am profoundly thankful to my beloved parents, whose unconditional love, sacrifices, and encouragement have been my greatest strength. Their unwavering belief in me has inspired me to strive for excellence.

Thank you all for being part of this meaningful chapter of my life.

## REFERENCES


Brown, A., et al. (2019). The Environmental Benefits of Smart Parking Systems. Environmental Science & Technology, 53(8), 4120-4130.

Brown, L., et al. (2021). "*GPU-Accelerated Real-Time Parking Monitoring System," Computing in Urban Transportation Journal*, vol. 29, no. 1, pp. 45–53.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Johnson, R., & Lee, S. (2020). "*Accuracy Assessment of Machine Vision in Parking Detection Systems," *Journal of Transportation Technologies*, vol. 11, no. 2, pp. 150–162.

Kim, T., & Roberts, M. (2020). "*Scalable Parking Detection with Machine Vision,*" *Urban Transportation Research Journal*, vol. 16, no. 5, pp. 200–212.

Kurose, J. F., & Ross, K. W. (2021). *Computer Networking: A Top-Down Approach*. Pearson.

Zhou, L., & Han, M. (2021). "Offline-first Mobile Applications: A Case Study Using SQLite for Resilient Data Handling." Journal of Mobile Systems, 12(3), 147–154.

National Parking Association. (2022). *Multi-Platform Compatibility in Parking Management Systems. Parking Today*, 15(2), 22-28.

National Parking Association. (2022). *State of the Parking Industry Report. Washington, DC: National Parking Association.*

Smith, J., & Lee, K. (2020). *Real-Time Parking Information Systems: Impact on User Satisfaction and Urban Traffic Efficiency. Journal of Urban Technology*, 27(3), 45-60.

Smith, J., et al. (2021). e *Parking Detection Using Convolutional Neural Networks," Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2340–2351.

Tanenbaum, A. S., & Van Steen, M. (2017). *Distributed Systems: Principles and Paradigms*. Pearson.

Xinhua News Agency. (2022, January 7). *Big data tells you where parking difficulties are more likely to occur*. https://news.qq.com/rain/a/20220107A0D1UX00

Zhang, Y., & Wang, H. (2019). "*Environmental Robustness in Machine Vision-Based Parking Detection*," *Journal of Advanced Transportation Systems**, vol. 17, no. 3, pp. 98–110.

Zhang, L., & Wang, H. (2021). Designing User-Friendly Interfaces for Smart Parking Applications. International Journal of Human-Computer Interaction, 37(5), 456-470.

Ultralytics. (2024). YOLOv8 Official Documentation. Retrieved from https://docs.ultralytics.com

Google. (2024). Firebase Realtime Database Documentation. Retrieved from https://firebase.google.com/docs/database

JetBrains. (2024). Jetpack Compose – Modern Toolkit for Building Android UI. Retrieved from https://developer.android.com/jetpack/compose

Google Maps Platform. (2024). Maps SDK for Android and Web. Retrieved from https://developers.google.com/maps

OpenCV. (2023). OpenCV Python Tutorials – Image Processing. Retrieved from https://docs.opencv.org/

PyTorch. (2023). An Open-Source Machine Learning Framework. Retrieved from https://pytorch.org

SQLite. (2023). SQLite Documentation. Retrieved from https://sqlite.org/docs.html

Firebase Admin SDK. (2024). Firebase Python Admin SDK Reference. Retrieved from https://firebase.google.com/docs/admin/setup