International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Smart Innovations in Computer Science and Applications
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

# An Intelligent Smart Parking Finder System Using Python and Real-Time Data Processing

Ananya Deshmukh
Department of Science and Technology, G.H.Raisoni Skill Tech University, Nagpur, India

ABSTRACT
The fast rise in vehicle density and the scarcity of parking facilities have made parking management in urban settings a major concern. Drivers frequently have trouble finding parking spots, which wastes time, uses more fuel, and causes traffic jams. Conventional parking systems lack effective information retrieval techniques and organised digital help.

In order to streamline the parking search process, this study discusses the design and development of a Smart Parking Finder System, a software-based lightweight solution. The system incorporates Python backend processing, a structured SQLite database for parking data management, and a graphical user interface (GUI) created using Tkinter. The program shows the available status in real time and lets users search parking spots based on car type and selected area.

The suggested prototype exhibits user-friendly functionality, quick reaction times, and effective database interaction. The system offers a useful and affordable strategy appropriate for academic demonstration and small-scale implementation. Real-time parking updates, GPS-based navigation, and Internet of Things integration are possible future improvements.

**Keywords:** Smart Parking System, Parking Management, Python Application, Tkinter GUI, SQLite Database, Urban Transportation, Parking Slot Availability, Software-Based Prototype, Offline Parking System, Database-Driven Search.



## 1. Introduction

Effective parking operation has come an decreasingly important concern in ultramodern civic surroundings. Rapid urbanization, population growth, and the rising number of vehicles have significantly increased the demand for parking spaces.

One of the major issues faced by motorists is the difficulty in locating available parking spaces. Traditional parking systems largely depend on homemade observation and warrant digital backing, performing in hamstrung application of parking coffers. motorists frequently spend considerable time searching for parking, which contributes to business traffic, energy destruction, and environmental pollution.

To address these challenges, smart parking results have surfaced that influence digital technologies to give structured parking information. These systems aim to ameliorate availability, reduce hunt time, and enhance overall parking effectiveness. still, numerous advanced results calculate on precious tackle factors similar as IoT detectors and real- time monitoring structure, which may not be doable for small- scale or prototype- grounded executions.

This exploration proposes a Smart Parking Finder System, a featherlight software- grounded operation designed to simplify the parking hunt process. The system provides a graphical stoner interface( GUI) that allows druggies to search parking locales grounded on area and vehicle type. By integrating Python for backend processing, Tkinter for interface design, and SQLite for database operation, the system offers a cost-effective and stoner-friendly result.

The primary ideal of this study is to design, apply, and estimate a functional prototype that demonstrates effective parking data reclamation, accurate vacuity display, and smooth stoner commerce.

### *1.1 Problem Statement*

Civic transportation systems are facing significant challenges due to the rapid-fire increase in vehicle power and limited parking structure. One of the most common issues endured by motorists is the difficulty in locating available parking spaces, particularly in densely peopled areas. The absence of structured parking information forces motorists to calculate on homemade searching, which results in inordinate time consumption, increased energy operation, and business traffic.

Traditional parking operation approaches lack intelligent mechanisms for furnishing real- time parking vacuity and effective hunt backing. motorists frequently encounter query regarding niche vacuity, leading to repeated searching cycles and hamstrung application of parking coffers. also, homemade parking systems do n't offer filtering options grounded on position or vehicle type, farther complicating the parking process.

The provocation behind this exploration arises from the growing challenges associated with parking operation in civic areas. With the nonstop increase in the number of vehicles, motorists constantly encounter difficulties in relating available parking spaces. This not only leads to frustration but also contributes to gratuitous energy consumption, business traffic, and environmental pollution.

In numerous metropolises, parking hunt remains a homemade and time- consuming exertion. motorists frequently spend a considerable quantum of time navigating through crowded thoroughfares to find suitable parking. The lack of structured digital backing and real- time vacuity information farther aggravates the problem.

Advancements in software technologies give an occasion to develop cost-effective parking operation systems without taking precious tackle structure. A featherlight operation able of reacquiring parking data, filtering results grounded on stoner preferences, and displaying vacuity status can significantly ameliorate the parking experience.

This exploration contributes to the design and development of a Smart Parking Finder System, a featherlight software- grounded prototype aimed at perfecting parking hunt effectiveness and stoner convenience. The study focuses on delivering a practical and cost-effective result that addresses common limitations of traditional parking systems.

The crucial benefactions of this work are epitomized as follows

- Development of a GUI- Grounded operation

A stoner-friendly graphical interface was designed using Tkinter to insure simple and intuitive commerce.

- Structured Database Design

An SQLite relational database was enforced to store and manage parking information efficiently.

- Rule- Grounded Parking Hunt Medium

A hunt functionality was developed to recoup parking details grounded on named area and vehicle type.

- Vacuity Display System

The system provides clear parking status information, enhancing decision- making for druggies.

- Featherlight and Offline Operation

The prototype demonstrates effective performance without taking internet connectivity or specialized tackle.

- Foundation for Future Enhancements

The system armature supports scalability for integration with IoT detectors, GPS modules, and intelligent vaticination ways.

## 1. Related Work

Parking operation has been an active area of exploration due to adding civic business traffic and hamstrung application of parking coffers. colorful smart parking results have been proposed to ameliorate parking vacuity discovery, reservation mechanisms, and stoner guidance systems.

Several studies have concentrated on IoT- grounded parking systems, where detectors are stationed to cover parking niche residency. These systems give real- time updates and reduce homemade intervention. still, the perpetration of IoT structure involves high installation and conservation costs, limiting feasibility for small- scale deployments.

Research has also explored pall- grounded parking operation platforms, enabling centralized data processing and remote availability. While pall integration improves scalability and monitoring capabilities, it introduces challenges related to network reliance, quiescence, and data security.

In addition, machine literacy approaches have been applied to prognosticate parking demand and optimize space application. These ways enhance system intelligence but bear large datasets, computational coffers, and complex training procedures.

Despite these advancements, numerous being results calculate on precious tackle factors or complex infrastructures. There remains a need for a featherlight, cost-effective, and software- driven prototype suitable for academic perpetration and demonstration purposes.

The proposed **Smart Parking Finder System** differs from previous works by emphasizing:

- Minimal hardware dependency
- Lightweight implementation
- Offline functionality
- User-friendly GUI interaction
- Structured database-driven search

## 2. Research Methodology

This section describes the systematic approach used for the design, development, and evaluation of the Smart Parking Finder System. The methodology outlines the problem definition, system development strategy, algorithm design, and validation procedures.

### *2.1. Problem statement*

Urban areas face increasing difficulties in managing parking spaces efficiently due to rising vehicle density and limited parking infrastructure. Drivers often spend excessive time searching for available parking slots, leading to traffic congestion, fuel wastage, and user inconvenience.

Traditional parking systems lack structured digital support and do not provide intelligent search or filtering mechanisms. The absence of organized parking information results in inefficient space utilization and uncertainty regarding slot availability.

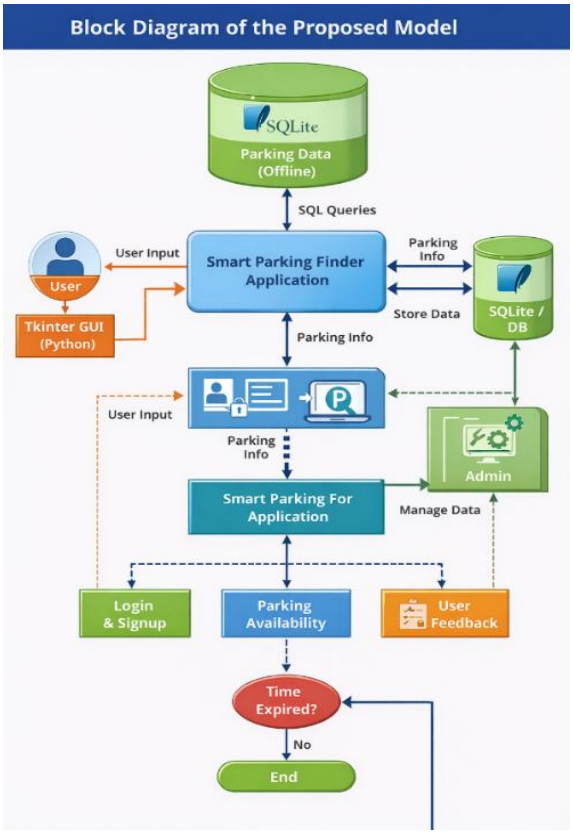


*Fig 1. Block diagram of the proposed model*

### *3.2 Proposed System / Model*

The proposed **Smart Parking Finder System** is a lightweight software-based application designed to simplify the process of locating available parking spaces. The system focuses on improving user convenience, reducing parking search time, and providing structured parking information through an intuitive graphical interface.

The system is developed using **Python** as the core programming language. A **Tkinter-based Graphical User Interface (GUI)** is implemented to enable interactive and user-friendly operation. Parking data is managed using an **SQLite relational database**, ensuring efficient storage and retrieval of records.

The application allows users to:

- Log into the system
- Search parking locations based on selected area
- Filter parking options by vehicle type
- View parking availability status

Unlike hardware-dependent smart parking solutions, the proposed system operates as a **software-driven prototype**, eliminating the need for IoT sensors or complex infrastructure. This approach ensures low implementation cost, lightweight performance, and offline usability.

### *3.3 System Design*

The Smart Parking Finder System follows a structured workflow to ensure efficient user interaction and accurate parking information retrieval. The workflow describes the sequence of operations performed by the system from user login to parking availability display.

Initially, the user launches the application and accesses the login interface. After successful authentication, the system redirects the user to the main dashboard. The user then selects the desired area/location and vehicle type. Based on the selected criteria, the system queries the SQLite database to retrieve relevant parking records.

The filtered results, including parking name, location, and availability status, are displayed on the graphical interface. This workflow ensures quick data processing, organized retrieval, and improved usability.

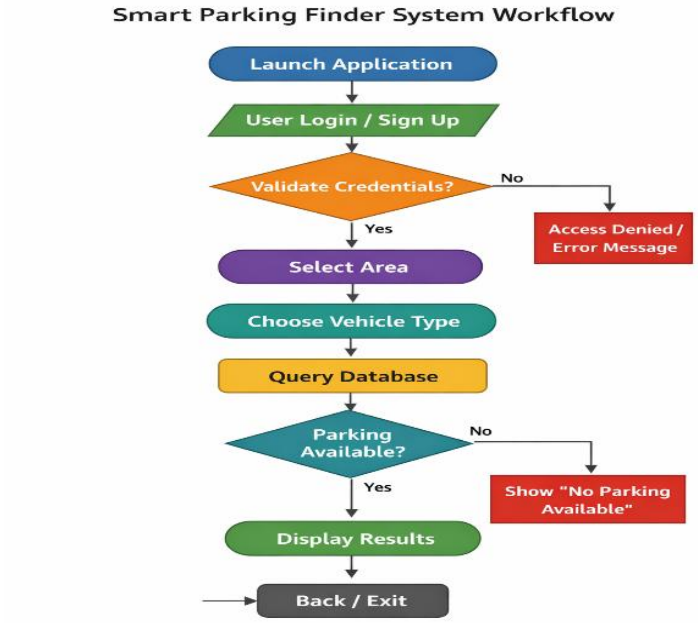


*Fig. 2. Smart Parking Finder System Workflow*

### *3.4 Proposed Algorithm*

The Smart Parking Finder System operates using a rule-based algorithm designed to ensure efficient parking search and accurate availability display. The algorithm defines the sequence of logical steps executed by the system based on user inputs.

The algorithm focuses on user authentication, input selection, database querying, and result presentation.

The algorithm begins with user authentication to ensure secure system access. Upon successful login, the user selects search criteria including area and vehicle type. The system processes the input parameters and executes structured SQL queries on the SQLite database. The retrieved data is filtered based on availability status and presented through the graphical interface.

### *Algorithm Steps*

| Step No. | Description |
|---|---|
| Step 1 | User launches the Smart Parking Finder application |
| Step 2 | System displays login interface |
| Step 3 | User enters login credentials |
| Step 4 | System validates user authentication |
| Step 5 | Dashboard interface is displayed |

| | |
|---|---|
| Step 6 | User selects area/location |
| Step 7 | User selects vehicle type |
| Step 8 | System queries SQLite database |
| Step 9 | Matching parking records retrieved |
| Step 10 | Parking availability displayed on GUI |

### *3.5 System Architecture*

The Smart Parking Finder System follows a modular and layered architecture to ensure efficient operation, maintainability, and scalability. The architecture is designed to separate user interaction, application logic, and data management into distinct functional layers.

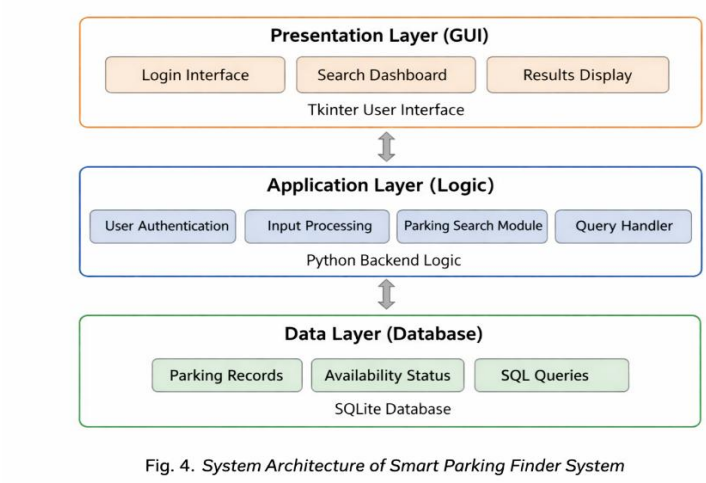


Fig. 4. *System Architecture of Smart Parking Finder System*

*Fig. 3. System Architecture of Smart Parking Finder System*

## 3. Implementation

This section describes the practical development and execution of the Smart Parking Finder System. It outlines the technologies used, system modules, database design, and interface implementation.

### *4.1 Development Environment*

The Smart Parking Finder System is implemented using a software-based development approach. The application is developed using:

- Programming Language: Python
- GUI Framework: Tkinter
- Database: SQLite
- Platform: Desktop-based execution

Python provides flexibility and rapid development capabilities, while Tkinter enables the creation of a lightweight graphical interface. SQLite is used for structured data storage and efficient query handling.

### *4.2 System Modules*

The system is divided into functional modules to ensure organized processing:

1. Login & Authentication Module

- Validates user credentials
- Ensures secure access

2. Input Selection Module

- Accepts area/location
- Accepts vehicle type

3. Parking Search Module

- Executes search logic
- Filters database records

4. Results Display Module

- Displays available parking slots
- Shows availability status

### *4.3 Database Implementation*

An SQLite relational database is designed to manage parking records.

Database stores:

- Parking Name
- Location
- Vehicle Type
- Availability Status

Structured SQL queries are used to retrieve matching parking data based on user input.

### *4.4 GUI Implementation*

The graphical interface is designed using Tkinter to provide ease of interaction.

GUI Screens:

**Login Screen** **Sign up Screen**

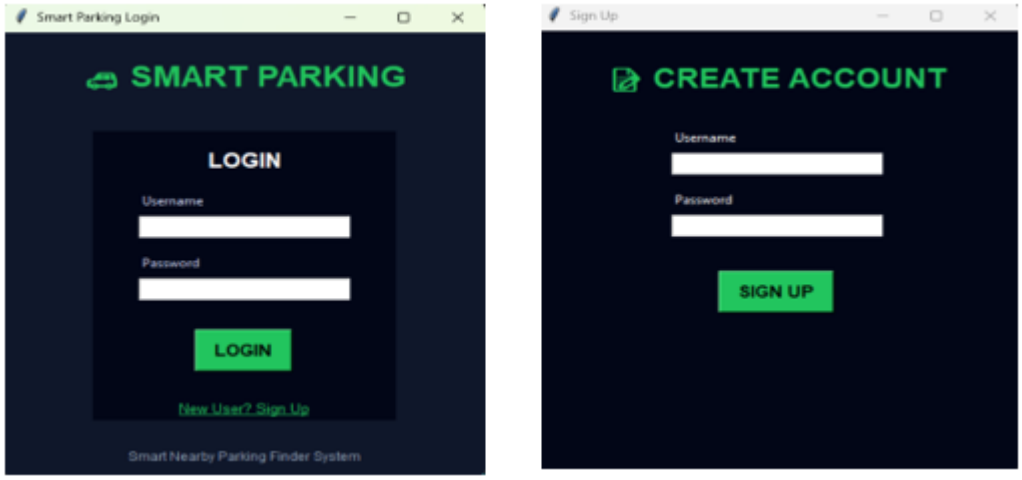


*Fig.4. Login Screen and Sign-up screen*

- **Smart Dashboard**

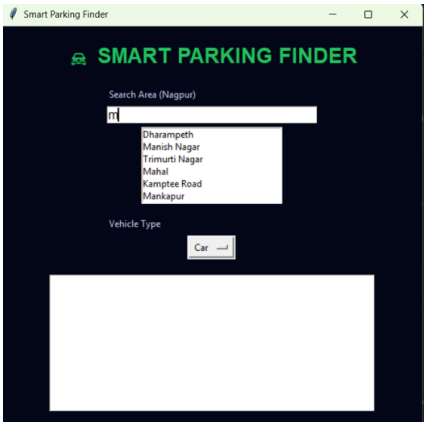


*Fig.5.Smart Dashboard*

### *4.5 System Execution Flow*

1. User logs into the system
2. User selects area and vehicle type
3. System queries database
4. Filtered parking results displayed

## *5. Results & Performance Evaluation*

This section presents the outcomes obtained from the implementation of the Smart Parking Finder System. The performance of the system is evaluated based on functional accuracy, response time, and usability.

### *5.1 System Testing*

The developed application was tested under multiple user scenarios to verify system functionality. Different combinations of input parameters such as area/location and vehicle type were provided.

Testing focused on:

- Correct login validation
- Accurate parking search results
- Proper GUI response
- Database query accuracy

The system successfully retrieved relevant parking records based on user inputs.

### *5.2 Functional Accuracy*

Functional accuracy represents the correctness of system outputs based on user inputs. The Smart Parking Finder System was evaluated across multiple functions including login validation, parking search, filtering mechanism, and availability display. The system demonstrated high reliability and consistent performance.

Functional accuracy refers to the correctness of the parking results displayed by the system.

- Correct parking locations displayed
- Vehicle-type filtering working properly
- Availability status shown accurately

The system demonstrated reliable performance in retrieving structured parking information.

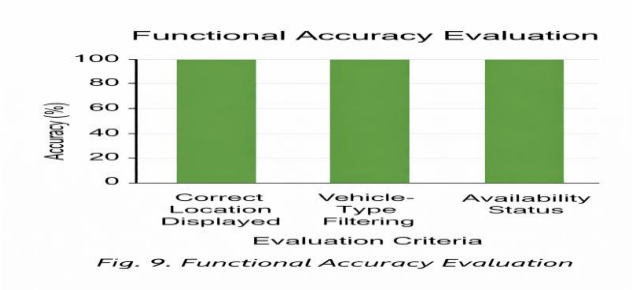


*Fig.6.Functional Accuracy Evaluation*

### *5.3 Response Time Analysis*

The response time of the Smart Parking Finder System was measured to evaluate system efficiency. The delay between user input submission and results display was analyzed. The system demonstrated minimal latency and smooth GUI updates.

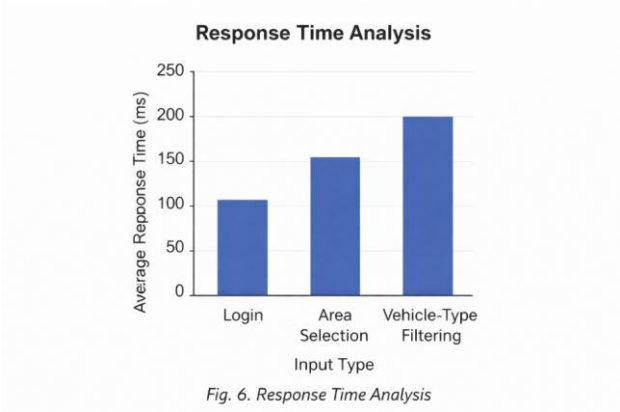


*Fig.7.Response Time Analysis*

### *5.4 Usability Evaluation*

Usability testing was conducted to evaluate user interaction quality.

Observations:

- Simple navigation
- Clear interface layout
- Easy input selection
- Readable output display

Users were able to operate the system without technical difficulty.

## 6. Conclusion and Future Work

### *6.1 Conclusion*

This research presented the design and development of a Smart Parking Finder System, a lightweight software-based application aimed at simplifying the parking search process. The system was developed using Python, Tkinter for graphical user interface design, and SQLite for structured database management.

The study addressed common challenges associated with traditional parking systems, including inefficient search mechanisms, lack of structured parking information, and user inconvenience. The proposed system enables users to search parking locations based on selected area and vehicle type while displaying availability status through an intuitive interface.

The implementation results demonstrate that the system provides:

- Efficient parking data retrieval
- Accurate availability display
- Fast response time
- User-friendly interaction

The prototype successfully validates the feasibility of a cost-effective and software-driven smart parking solution suitable for academic demonstration and small-scale applications.

### *6.2 Future Work*

Although the developed system performs effectively as a prototype, several enhancements can be incorporated to improve functionality and scalability:

IoT Integration:

Real-time slot occupancy detection using sensors

GPS Navigation:

Location-based parking guidance

Real-Time Availability Updates:

Dynamic database synchronization

Mobile Application Development:

Cross-platform accessibility

Machine Learning Integration:

Predictive parking availability

### *References*

1. G. Shaheen and E. Cohen, “Smart Parking Systems: A Review of Technologies and Applications,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1025–1035, 2020.
2. S. Kumar and R. Singh, “IoT-Based Smart Parking Management System,” *International Journal of Computer Applications*, vol. 182, no. 12, pp. 25–30, 2019.
3. A. Yadav and P. Sharma, “Design and Implementation of a Smart Parking System Using Wireless Sensor Networks,” *Procedia Computer Science*, vol. 125, pp. 1–8, 2018.
4. M. Patel and N. Shah, “Cloud-Based Smart Parking Solutions for Urban Areas,” *Journal of Emerging Technologies*, vol. 7, no. 2, pp. 45–50, 2021.
5. R. Gupta, “Database Management Techniques for Smart City Applications,” *International Journal of Data Engineering*, vol. 10, no. 4, pp. 112–118, 2022.
6. A. Chaube, "ACO-Enhanced Siamese Networks for Robust Feature Matching in Copy-Move Image Forgery Detection," 2024 International Conference on Artificial Intelligence and Quantum Computation-Based Sensor Application (ICAIQSA), Nagpur, India, 2024, pp. 1-6, doi: 10.1109/ICAIQSA64000.2024.10882433.
7. Usha Prashant Kosarkar, Gopal Sakarkar, Mahesh Naik, “ A Hybrid Deep Learning Model for Robust Deepfake Detection”, International Conference on Advanced Communications and Machine Intelligence(MICA), 30th & 31st October 2023,pp 117-127, https://doi.org/10.1007/978-981-97-6222-4_9