

Article

Mobile Application for Online Control and Booking of Free Parking Spots

Simona Filipova-Petrakieva

Department “Fundamentals of Electrical Engineering”, Faculty of Automation, Technical University of Sofia, blvd. Kl. Ohridski 8, 1000 Sofia, Bulgaria; petrakievas-te@tu-sofia.bg; Tel.: +359-876-765-431

Abstract

Under conditions of increasing traffic and limited parking options, mobile apps for real-time monitoring and booking of available spots in public parking lots are becoming an indispensable tool for making our daily lives easier. These apps offer convenience, save time, and reduce the hassles associated with finding a parking spot, while also contributing to a more efficient and organized urban space. Additionally, they help reduce traffic congestion and lower pollution levels by encouraging users to make more rational use of available parking resources, which benefits all city residents. This article describes the development of a mobile application for real-time booking of parking spots. The proposed application integrates useful features from existing mobile apps, adds its own new features, and is adapted to the conditions of life in Bulgaria. Its main advantage is that it is available completely free of charge. It offers the following functionalities: booking or recommending parking spots; checking parking spot availability; and providing feedback on current and past parking spot bookings, including ratings and comments. The application's interface is intuitive, which makes it easy to use. The following technologies were used in its development: Java, Android Studio, XML, Gradle, Android SDK, and Firebase. The application was compared to similar ones, incorporating their best features and adding new ones that improve upon them. The current work aims to help Bulgarian users save time and money spent looking for parking spots.

Keywords: online monitoring and booking lots in parking spots; Model–View–Presenter (MVP); Java; Android Studio; XML; Firebase

Academic Editors: Artur Kierzkowski, Tomasz Nowakowski, Agnieszka A. Tubis, Franciszek Restel, Tomasz Kisiel, Anna Jodejko-Pietruczuk, Mateusz Zająć, Viktoria Ivannikova, Michał Stosiak and Andrija Vidović

Received: 7 June 2026

Revised: 16 July 2026

Accepted: 16 July 2026

Published: 20 July 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

The continuously growing urban population significantly increases demand for efficient transportation services, a trend that is linked to the development of smart urban infrastructure. The rise in the number of private cars has led to traffic congestion, a shortage of parking spaces, environmental pollution, and increased fuel consumption. This has made solving the parking problem one of the main challenges facing modern cities. As a result, digital technologies are increasingly being used to improve urban mobility and enhance the quality of transportation services.

In recent years, the idea of “smart cities” has been proposed as a comprehensive framework for addressing these challenges by integrating information and communication technologies, the “Internet of Things” (IoT), cloud computing, big data analytics, and intelligent decision-support systems into urban infrastructure. The main goal is to opti-

mize the use of urban resources while improving citizens' quality of life through interconnected digital services [1,2].

One of the key components of "smart cities" is "smart mobility," which focuses on providing safe, efficient transportation services that make users' daily lives easier. "Smart mobility" integrates intelligent transportation technologies, mobile apps, connected infrastructure, and real-time information systems to optimize the movement of people and vehicles, while reducing traffic congestion, travel time, fuel consumption, and the environmental impact of harmful exhaust emissions [2,3,4]. In this context, smart parking management has become an indispensable service, as the search for available parking spaces contributes significantly to eliminating unnecessary urban traffic and reducing vehicle emissions.

Intelligent Transportation Systems (ITSs) are the foundation of smart mobility. They combine sensor technologies, wireless communication, data processing, and intelligent control algorithms to improve transportation efficiency and safety. ITS applications include traffic monitoring, route navigation, passenger information systems, electronic payments, fleet management, and smart parking services [5,6]. The rapid development of Internet of Things (IoT) technologies, cloud computing, wireless sensor networks, and mobile computing has further accelerated the development of modern ITS applications, which are now capable of processing big data in real time [7,8].

Among the many services in the field of ITS, intelligent parking systems have generated considerable research interest over the past decade. Their main goal is to help drivers find, book, and manage parking spaces more efficiently, while reducing unnecessary vehicle traffic. Modern smart parking solutions typically combine IoT devices, wireless sensor networks, Radio Frequency Identification (RFID) technology, camera surveillance, cloud platforms, and mobile apps to detect parking space occupancy and provide real-time parking information [3,9,10].

Recent developments in mobile applications have expanded the functionality of conventional parking reservation systems by introducing mechanisms to book parking spots before arriving at the parking spot. These systems use optimization algorithms to allocate parking spots based on multiple criteria, such as estimated time of arrival, travel distance, parking duration, pricing strategies, user preferences, and parking spot utilization efficiency [5,11]. Such parking systems significantly improve the utilization of parking resources, reduce search time, alleviate traffic congestion, and increase user satisfaction compared to traditional First-In, First-Out (FIFO) queue-based approaches [3,11].

Despite the significant progress made in recent years, some limitations still exist. Existing smart parking solutions are often developed for specific cities or parking lots, which limits their scalability and interoperability. Furthermore, many published surveys focus primarily on detecting available parking spaces, rather than providing comprehensive reservation platforms that integrate user registration, parking space reservation, reservation management, navigation, and real-time communication within a single mobile application [4,9,10,12]. In addition, several proposed systems require expensive sensor infrastructure or the widespread deployment of physical sensors. This increases the costs of their implementation and maintenance [3,9].

The reviewed literature also shows that although IoT, cloud, and mobile technologies have significantly improved the functionality of smart parking systems, this is not sufficient, and further research is needed to develop integrated, scalable, cost-effective, and user-friendly parking reservation platforms to support sustainable urban mobility [2,4,7,8,11,12].

Based on the challenges mentioned above, a mobile system for smart parking space reservations, designed to support effective parking management in Bulgaria, is presented. The proposed solution combines mobile computing technologies with cloud services to

provide real-time parking space reservations, reservation management, and user-oriented parking assistance, thereby contributing to the development of smart transportation services within the “smart city” ecosystem.

The rest of this paper is organized as follows. In the next section, similar mobile applications are analyzed and compared to each other. The architecture of the developed mobile application is presented in Section 3. Its software implementation is described in Section 4. The results of its workability are presented in Section 5. A discussion about the functionalities of the suggested mobile app compared with similar ones is presented in Section 6. User evaluation and validation is described in Section 7. The paper finishes with concluding remarks about the functionalities and benefits provided by the app.

2. Related Apps for Parking

There are many mobile apps for monitoring and booking parking spots. We will focus on a few of them that combine basic essential features.

Parkopedia [13] is a platform created in the UK that provides information on parking locations around the world. It is available for Android, iOS, and the web and allows users to search for parking spots, view prices, and book spots in real time. Although the app includes extremely useful features, the interface is complex, which makes it difficult to learn how to use. This results in an average level of usability.

ParkMe [14] is a platform developed in the United States. It is available for Android and iOS operating systems. It allows users to view pricing for the parking spots they are considering and includes a built-in navigation feature to guide them to available parking spots. The app features an intuitive user interface, making it convenient for quickly finding and booking a parking spot.

JustPark [15] is a platform developed in the UK. Like Parkopedia, it is available for Android, iOS, and the web. The app allows users to book parking spots and manage them. Its interface is intuitive and user-friendly. Despite its rich set of features, JustPark is better suited for those looking for advance bookings and long-term parking rather than for booking parking spots for short periods of time.

EasyPark [16] is a platform developed in Sweden. It is available for Android and iOS operating systems. It allows users to navigate to the desired parking spot and pay the fee online. Depending on the region of use, this app offers varying degrees of convenience due to regional differences in fees and available features. Overall, it can be considered user-friendly for users in Europe.

After analyzing the functionality of the examined mobile parking applications, it can be observed that most existing solutions primarily focus on parking discovery, navigation, or payment services. Although these applications provide valuable information about parking facilities, their functionality for parking reservation is often limited or unavailable, and their geographical coverage depends on the supported regions and parking operators.

To address these limitations, the proposed mobile application, Parky, was designed as a reservation-oriented parking management system. Unlike conventional parking search applications, Parky integrates parking search, real-time parking space reservation, availability management, and navigation within a single platform. The application provides detailed information about each parking facility, including parking availability, service fees, contact information, and direct communication options. In addition, it maintains a history of previous reservations and recommends suitable parking locations according to user-selected criteria.

Another important contribution of the proposed solution is its adaptation to the operational characteristics of the Bulgarian parking infrastructure, where comprehensive reservation services are still limited. At the same time, the system architecture has been

designed to be scalable and can be extended to support additional parking operators, IoT-based occupancy detection, and other smart city services.

3. Architecture of the Developed Mobile App

3.1. Base Architecture

The proposed Parky mobile app is based on a three-layer architecture, Model–View–Presenter (MVP) (Figure 1), which makes it suitable for development using Android Studio.

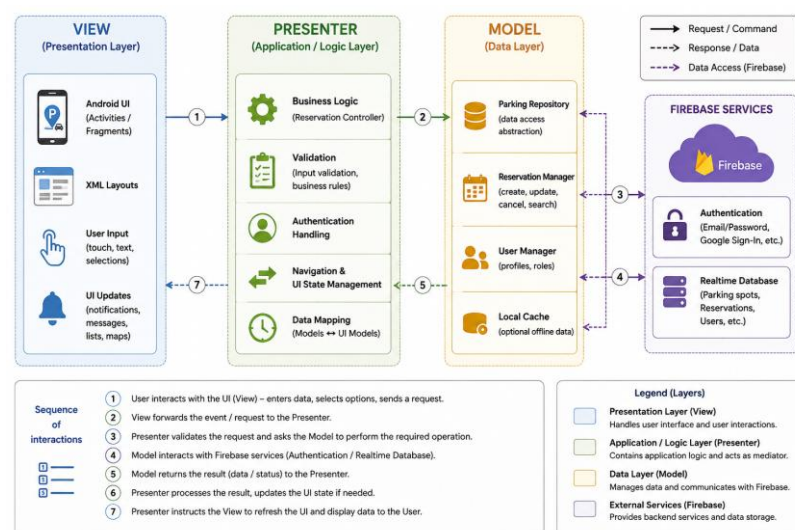


Figure 1. Model–View–Presenter (MVP) architecture.

The MVP model is a software design approach that divides program logic into three main components: Model, View, and Presenter [17–19]. This ensures a clear separation between data processing, information visualization, and control logic.

- **Model:** This controls the data, structures, and internal logic of the application. In Parky, the model is responsible for processing and storing data related to users, parking spots, and books. Data is synchronized in real time via the Firebase database, ensuring consistency and security.
- **View:** This represents the visualization of the information and the application's user interface. In Parky, this is implemented using XML, which enables the visualization of login and registration forms and user profiles, searching for and booking parking spots, and other functionalities. The user interface is responsible for ensuring a pleasant user experience.
- **Presenter:** This receives input data from the user and converts it into commands for the Model or View. In Parky, the Presenter controls the interactions between the user and the application by processing user actions (such as booking and parking recommendations) and ensuring that the View is updated in response to changes in the data.

The three-layer MVP architecture used in Parky ensures flexibility and scalability in the application's functioning, along with easy support and functional upgrades. The separation of the logic into three main components (Model, View, and Presenter) ensures that each part of the system can be developed and improved independently of the others, contributing to better stability and optimization of resource usage.

3.2. Functions Implemented

In the developed mobile application Parky, the following functions are implemented:

Registration: The user can create a new account in the system by entering the required registration information.

Login: After registration, the user can log in to the system using their login details (username and password).

Feedback: The user can provide feedback on the app, including ratings and comments, which will help improve the service.

Booking or recommending parking spots: The user can book or recommend a parking spot. This includes checking whether they have already booked or recommended the same parking spot.

Checking for vacancies: Check whether a parking spot has already been booked or recommended. This function checks whether the user has already booked or recommended a specific parking spot to avoid duplicate actions.

Log out: The user can log out of their account and return to the login screen.

View my bookings: The user can view a list of all their current and past parking spot bookings.

View recommended parking spots: The user can view a list of parking spots that have been recommended, including those recommended by themselves or other users.

The process of booking a parking spot is visualized in the diagram shown in Figure 2.

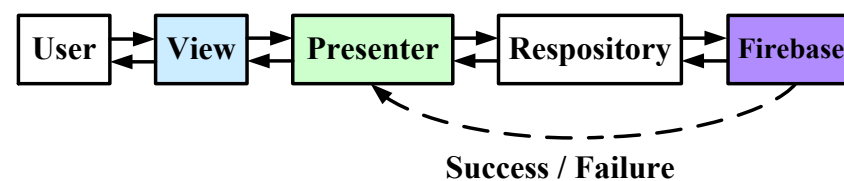


Figure 2. Sequence diagram for parking spot booking.

The sequence of steps for booking parking spots is as follows.

- **User:** The user initiates the process by submitting a booking request (Request Booking).
- **Frontend:** This receives the request from the user and sends an HTTP POST request to the server (FindParkingDetailsActivity) with the path “/reserve”.
- **FindParkingDetailsActivity:** This is responsible for processing requests. It calls the checkAvailability() method in the data model (ParkingSlotModel) to check the availability of parking spots.
- **ParkingSlotModel:** This processes the availability check request and returns a response to FindParkingDetailsActivity, indicating whether or not there are available spots in the parking lot.
- **FindParkingDetailsActivity:** Based on the response from ParkingSlotModel, FindParkingDetailsActivity calls the saveReservation() method in the Firebase database to save the booking if spots are available.
- **FirebaseDatabase:** This stores the booking information and returns a response to FindParkingDetailsActivity, indicating whether the operation was successful or not.
- **FindParkingDetailsActivity:** After receiving a response from FirebaseDatabase, FindParkingDetailsActivity returns the result (TRUE or FALSE) to the frontend.
- **Frontend:** It returns the result and displays the booking status to the user (Show Reservation Status).

4. Software Implementation

Android Studio was used to develop the frontend and backend components of the Parky mobile app. This powerful integrated development environment specializes in creating mobile apps for the Android platform and offers extensive capabilities for writing and testing code on both the frontend and backend. Android Studio also offers debugging and profiling tools that simplify the identification and resolution of bugs, as well as the optimization of the app's performance. This development environment was key to the successful creation of Parky, providing a comprehensive platform for controlling all aspects of the app's development [20–22].

Extensible Markup Language (XML), which is the standard language for defining the user interface in Android applications, was used to develop the frontend part of the Parky application. XML [23] provides an easy and declarative way to create the visual elements that make up the application's interface. By using it, developers can create complex layouts and components that meet the needs of users. The main components of the frontend are as follows:

- **ConstraintLayout:** This is one of the main components used in the application. It is used to arrange visual elements. It enables flexible and efficient positioning of elements, allowing constraints to be defined between them. This results in an adaptive and dynamic user interface. It also simplifies the support and expansion of the application by ensuring that images are resized on different devices with varying screen sizes.
- **ImageView and TextView:** These are the main display components in the app. ImageView is used to display graphical resources such as images of parking spots or icons, while TextView is used to display text information such as parking spot names, contact information, and other details. The LinearLayout feature is used to arrange multiple elements vertically or horizontally. This ensures a clear structure and consistency in the layout of the content on the screen.
- **Menu XML files** define the navigation options available to the user within the app. They contain various navigation elements such as profile, feedback, password change, parking search, and others, providing quick access to the app's main functions. The menu is easy to customize and offers convenient and intuitive navigation.
- **AppCompatButton:** This is a function for creating buttons. It allows users to make recommendations regarding parking, bookings, and navigation.

The Java programming language was used to develop the backend of the Parky application to implement business logic and data processing [24]. Java provides a powerful platform for creating stable and efficient applications, with extensive capabilities for data control and user interface interaction. The main components of the backend are as follows:

- **Business logic and data control with Java:** Java is used to implement the application's core business logic, such as processing bookings, customer data control, parking recommendations, and navigation. Java classes are responsible for processing user input, interacting with the user interface, and performing operations such as monitoring available parking spots and creating notifications.
- **Users' sessions control:** The connection to Firebase is implemented in Java using Firebase Authentication (FirebaseAuth), which provides secure access to the application and control of user sessions. Through FirebaseAuth, users can register, log in, and maintain a secure session while interacting with the application.
- **Data storage and real-time synchronization:** The Firebase Realtime Database is integrated into the application via Java to provide data storage and data control, such as parking spot information, bookings, recommendations, and user data. Any changes to the data are synchronized in real time between client devices and the Firebase da-

tabase, ensuring that users always have access to up-to-date information. Java classes handle communication with Firebase and perform the necessary operations for updating and extracting data.

- Event processing and notifications: Java is also responsible for processing user events and generating in-app notifications. For example, when a user books a parking spot, Java classes update the relevant data in Firebase and generate in-app notifications to inform the user that the operation was successful.
- Connectivity control: The app uses Java to control Internet connectivity and check for an active connection before performing operations that require communication with Firebase. This is an important aspect of ensuring the app works reliably and correctly, especially for operations that require access to real-time data.

User data is stored in a real-time Firebase relational database, which plays a key role in ensuring that data is stored correctly and synchronized in real time [25].

The operations shown in Figure 2 are implemented using the software described above through the following functions:

- Request Book: Specify the starting point where the user submits a booking request.
- HTTP POST/book: The request sent from the frontend to FindParkingDetailsActivity to start the booking process.
- checkAvailability(): A function in the data model that checks whether parking spots are available.
- saveReservation(): A function that saves the booking in the database if available spots exist.
- Show Book Status: The final step, where the booking confirmation is sent to the user.

The Parky app uses the Firebase relational database, which works in real time, for data storage and processing. Firebase is a cloud-based database (DB) that stores data in JSON format and synchronizes information in real time with all clients connected to the database. This ensures immediate updates and fast access to data, regardless of the number of connected users. The primary method for linking data is through the use of users' unique identifiers (user IDs).

The DB in Parky is organized as follows:

1. MainPages: This section contains data on the main pages in the app, such as the password reset and feedback.
2. ParkingSlots: This part of the DB contains information on all available parking spots. The following data is stored for each parking spot:
 - Contact: The phone number to contact the parking spot.
 - Email: The user's email address associated with a specific parking spot.
 - Latitude and Longitude: The geographic coordinates of the parking spot, used for map display and navigation.
 - Name: The name of the parking spot.
 - ParkingImage: An image associated with the parking spot, displayed in the user interface.
 - Prices: The pricing information for the parking spot.
 - SelectedPrice: The price selected by the user to reserve the parking spot.
 - Status: The current status of the parking spot (e.g., "Available").
 - UsedId: The unique ID of the user who reserved the parking spot.
3. Users: This section stores information about the app's registered users. For each user, a unique identifier and personal data—such as email address, name, age, vehicle type, vehicle model, and registration number—are stored and used to control profiles and reservations.

The relationships among the data in the individual sections are as follows:

- **Users and ParkingSlots:** The relationship between these two entities allows parking spots to be linked to specific users, such as by reserving selected or recommended parking spots for a given user.
- **Users and Feedback:** Every piece of feedback or rating provided by a user is linked to a specific user through the user ID. This allows for the personalization of the collected feedback.
- **Users and ChangePassword:** Every password change is linked to a specific user, with information about the old and new passwords stored by the application.

Main functionalities of the Firebase Realtime DB, realized in the suggested application:

1. Real-time storage and control of big data, even with a large number of users.
2. Fast access to and processing of information that is updated and synchronized in real time.
3. Protect data through built-in access control mechanisms.
4. Database scalability as the number of users increases.
5. Statistical processing of data in the database, such as the number of active users per day, helps analyze user behavior and optimize the application's performance.
6. Ability to integrate with other Google services and platforms, simplifying the expansion of the application's functionalities.

User data security is a key priority in Parky. All data, including user passwords, is protected through hashing. The SHA-256 hashing function is used to encrypt passwords. It is part of the SHA-2 family of algorithms, which guarantee high security and data resilience against hacker attacks. The main features provided by this function are:

- A high level of security and resilience against hacker attacks, since once a password is hashed, the original password cannot be easily recovered from the generated hash code.
- Every time a password is changed, both the current and the new passwords are hashed before being stored in the database. This ensures password protection even in the event of unauthorized access to the database.
- Access to the database is restricted and protected using a unique user identifier (user ID) to link user data to their identity. This provides an additional layer of security and control over access to information.

The application also uses all mechanisms built into the Firebase database for controlling and protecting data from unauthorized access, such as Firebase Authentication, Firebase Security Rules, Role-Based Access Control (RBAC), Custom Claims, Cloud Firestore Security Rules, Realtime Database Rules, Cloud Storage Security Rules, Firestore App Check, HTTPS/TLS Encryption, Encryption of Data at Rest, Firebase Admin SDK, Audit Logging (by Google Cloud), Data validation, Ownership-based Access Control, and Least Privilege Principle.

The proposed DB structure supports real-time system operations. The relationships between users and various DB components—such as feedback and password changes—enable effective monitoring and control of actions within the application. This flexible structure ensures the application's security and scalability.

5. Results of Workability Tests

To visualize and test the developed application, both Android emulators and physical devices are used to ensure compatibility and comfortable user experience across different device screen sizes and operating system versions.

The database provides a reliable and scalable backend service that synchronizes data between clients and the server. The security of user data against hacking attacks on the Parky app is ensured through password hashing. The generated hash code cannot be easily decoded back to the original password, which guarantees a high level of protection. For every operation involving a password change, both the current and new passwords are hashed before being stored in the database. This ensures that even if the database is compromised, users' actual passwords remain protected. Furthermore, access to the database is restricted and secured through the use of a unique user ID to link user data to their identity. This provides an additional layer of security and control over access to sensitive information.

The functionality of the Parky mobile app has been verified through testing. For the app to function properly, the device's GPS module is required to monitor the user's current location and provide information about the nearest available parking spots. A constant Internet connection is required to store and synchronize information on available parking spots located near the user.

The results of the tests performed are summarized in Table 1.

Table 1. Functionalities and features of the application.

Nº	Functionalities	Benchmark Test	Comparison Criteria	Status	Bugs
1	The app serves as an entry and exit for the users.	Test 1	Check whether the user successfully logs in to the system when entering the correct data.	Passed the tests.	-
		Test 2	Check whether an error is generated when invalid data is entered.		
		Test 3	Check whether, when the exit button is pressed, the users are logged out of the app and their session is terminated.		
2	The app sends feedback.	Test 4	Check whether the user can submit feedback through the app.	Passed the test.	-
3	The app books a parking spot.	Test 5	Check whether the user can book a parking spot through the app.	Passed the test.	-
4	The app recommends a parking spot.	Test 6	Check whether the user can recommend a parking spot through the app.	Passed the test.	-
5	The app both recommends and books parking spots.	Test 7	Check whether the user can both book and recommend a parking space.	Passed the test.	-
6	The app visualizes recommended parking spots.	Test 8	Check whether the app visualizes the recommended parking spots to the user.	Passed the test.	-

For each test, the process was as follows:

- Tests 1, 2, 3: This feature provides the application's base functionality related to user authentication. Logging in requires entering valid credentials, and logging out terminates the active session.
- Test 4: This test verifies whether the user can successfully submit feedback through the app. A successful test indicates that communication between users and the support team is functioning properly.

- Test 5: The system checks whether the user can successfully book a parking spot. Passing this test indicates that the app is communicating correctly with the database and updating the available parking spots.
- Test 6: The system checks whether the user can recommend a parking spot. A successful test means that the recommendation function works properly and that data is stored and shared correctly.
- Test 7: The system checks whether a user can both book and recommend a parking spot. Successful completion of this test indicates that the system can perform both actions without any conflicts.
- Test 8: The test verifies whether the app displays the recommended parking spots to the user. Passing this test indicates that the app correctly displays and filters the recommendations.
- Test 9: The test checks whether the user can view the history of all their previous bookings. A successful test confirms that the app correctly stores and retrieves data on previous bookings.

All benchmarks were passed successfully, which means that the developed mobile application functions correctly in accordance with the specified requirements and that no bugs were found in its implementation.

6. Discussion

Table 2 presents a comparative analysis of the Parky app and existing platforms for monitoring and booking parking spots that are similar and widely used.

The comparison was based on the following criteria: the ability to book a parking spot online, navigation to available parking, and the provision of information on prices and operating hours for each parking spot. The main differences between these platforms lie in their coverage, ease of use, intuitiveness of the user-friendly interface, and availability of additional functions, such as recommendations from other users and information regarding parking spot safety. The table also provides information on the country for which the respective app was developed; the operating system it runs on; and its price.

Table 2. Functional comparison of existing parking applications and the proposed Parky system.

Feature	Parky	Parkomedia	ParkMe	JustPark	EasyPark
Search for parking facilities	✓	✓	✓	✓	✓
Parking reservation	✓	✓	✗	✓	✗
Recommendation of parking spots	✓	✗	✗	✗	✗
Parking availability information	✓	✓	✓	✗	✗
Navigation to parking location	✓	✓	✓	✗	✓
Parking fee information	✓	✗	✓	✗	✓
Reservation history	✓	✗	✗	✗	✗
Parking ratings	✓	✗	✗	✗	✗
Contact information	✓	✗	✗	✗	✗
Direct phone call	✓	✗	✗	✗	✗
Cloud database synchronization	✓	✗	✗	✗	✗
Designed for Bulgarian parking infrastructure	✓	✗	✗	✗	✗
Integrated parking management platform	✓	✗	✗	✗	✗

The comparison presented in Table 2 shows that existing parking applications mainly focus on individual functionalities, such as parking search, navigation, online payment, or reservation services. Although these applications provide useful support for drivers,

they typically offer only a subset of the functionalities required for comprehensive parking management.

In contrast, the proposed Parky application integrates several complementary functionalities within a single mobile platform. In addition to parking search and reservation, the application provides recommendation-based parking selection using user ratings, parking availability information, reservation history, detailed parking information, direct communication with parking operators, and cloud-based synchronization of parking data. This integration enables users to complete the entire parking process—from selecting an appropriate parking facility to managing reservations—without relying on multiple independent applications.

Another important feature of the proposed solution is its adaptation to the operational requirements of the Bulgarian parking infrastructure, where integrated parking reservation services are still limited. While the current implementation has been developed for parking facilities in Bulgaria, the software architecture is modular and scalable, enabling future extension to additional parking operators, geographical regions, and smart city services.

7. User Evaluation and Experimental Validation

7.1. Evaluation Methodology

The usability and functional suitability of the proposed Parky mobile application can be evaluated through a controlled user study that combines subjective user ratings with objective task-performance indicators. The subjective assessment was based on a five-point Likert scale [26], where 1 = Strongly disagree, 2 = Disagree, 3 = Neither agree nor disagree, 4 = Agree, and 5 = Strongly agree. The purpose of the study is to assess whether users can learn and operate the application easily, complete the main parking-related tasks efficiently, understand the displayed information, and perceive the integrated reservation and recommendation functions as useful. The evaluation should be performed only after each participant has completed the same predefined set of tasks. To ensure safety and consistency, the test should be conducted in a laboratory, office, or simulated environment while the vehicle is stationary.

7.2. Participants and Sampling

The participant group should consist of licensed drivers (category B) who use Android smartphones and they are described in Table 3. For an initial usability study, a sample of 30 participants has been tested. Such a sample should be described as diverse rather than nationally representative unless a formal probability-based sampling procedure is applied. The demographic structure of the sample should be reported transparently so that the scope and limitations of the findings are clear.

Table 3. Participants' profile.

Characteristic	Categories
Age	18–29; 30–44; 45–59; 60+
Place of residence	Sofia; Plovdiv; Varna; Burgas; Blagoevgrad
Driving experience	Less than 2 years; 2–10 years; more than 10 years
Driving frequency	Daily; several times per week; occasionally
Smartphone proficiency	Basic; Intermediate; Advanced
Previous use of parking applications	Yes; No

7.3. Test Procedure

All participants should perform the following tasks: log in to the application; search for a parking facility in a selected city; open detailed information for a parking facility; select a parking facility based on recommendations or ratings; reserve a parking space; review the reservation history; and start navigation to the selected parking facility. If the current implementation supports reservation modification or cancellation, this task should also be included. During the test, the researcher should record whether each task is completed successfully, the time required, the number of errors, and the number of requests for assistance.

7.4. Asked Questions

After completing the tasks, participants should evaluate Parky using the statements presented in Table 4. The questionnaire measures learnability, interface clarity, effectiveness, efficiency, information quality, usefulness of recommendations, functional integration, confidence, intention to use, and overall satisfaction. One negatively worded item is included as a consistency check and must be reverse-scored during the analysis.

Table 4. Likert-scale questionnaire.

No	Statement	Aspect
Q1	The application was easy to learn.	Learnability
Q2	The main functions were easy to find.	Navigation and interface clarity
Q3	The information displayed on the screen was clear and understandable.	Understandability
Q4	The text, icons, and buttons were easy to read and recognize.	Visual clarity
Q5	I was able to search for a parking facility without assistance.	Effectiveness
Q6	The parking reservation process was clear and straightforward.	Task efficiency
Q7	I completed the reservation process in a reasonable amount of time.	Efficiency
Q8	The information about availability, fees, and location was sufficient.	Information quality
Q9	The recommendations and user ratings were useful when selecting a parking facility.	Recommendation usefulness
Q10	The navigation function helped me reach the selected parking facility.	Functional usefulness
Q11	The reservation history was useful for managing bookings.	Reservation management
Q12	The application functions were well integrated.	Functional integration
Q13	I felt confident while using the application.	User confidence
Q14	I would use Parky when searching for and reserving parking.	Intention to use
Q15	Overall, I was satisfied with the application.	Overall satisfaction
Q16	The application was unnecessarily complicated to use.	Reverse-scored item

7.5. Statistical Analysis

The subjective ratings obtained should be complemented by objective indicators, including task completion rate, time required to search for a parking facility, time required to complete a reservation, number of user errors, number of requests for assistance, and number of screens or interaction steps required. For each Likert item, the mean value, standard deviation, and percentage of positive responses (scores of 4 and 5) should be calculated. The reverse-scored item should be transformed before calculating any overall score. If several questionnaire items are combined into a common usability construct, internal consistency may be examined using Cronbach's alpha. The objective and subjective results should be interpreted together because high satisfaction alone does not prove efficient task performance, while rapid task completion does not necessarily imply a positive

user experience. The evaluation results obtained from the responses to questions Q1 through Q16 are summarized in Table 5.

Table 5. Evaluation results.

Evaluation Criterion	Related Question(s)	Mean	Standard Deviation	Positive Responses, % (scores 4 and 5)
		$\bar{X} = \frac{\sum_{i=1}^n x_i}{n}$	$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{X})^2}{n-1}}$	
Learnability	Q1	4.95	0.28	98.7
Navigation	Q2, Q10	4.78	0.35	94.6
Interface clarity	Q3, Q4	4.87	0.41	96.1
Parking search	Q5	4.93	0.56	95.3
Reservation process	Q6, Q7	4.98	0.64	97.8
Information quality	Q8	4.48	0.80	52.3
Recommendation usefulness	Q9	4.76	0.74	82.1
Reservation management	Q11	4.82	0.68	79.7
Functional integration	Q12	4.88	0.11	97.5
User confidence	Q13	4.76	0.51	78.3
Intention to use	Q14	4.83	0.25	97.9
Overall satisfaction	Q15	4.95	0.28	99.1

Criteria Q16 from Table 4 is a reverse-scored control item and should be used only for reliability analysis (e.g., Cronbach's alpha), not for the evaluation criteria. For this reason, it is not included in Table 5.

A convenience sample may provide valuable evidence for initial validation but does not support generalization to all Bulgarian drivers. The results may also be affected by differences in smartphone experience, familiarity with parking applications, and the artificial nature of a test group. These limitations should be reported explicitly. Future studies may extend the evaluation by involving larger and more geographically diverse samples, comparing Parky with alternative applications under the same task conditions, and conducting field trials in real parking environments.

The methodology applied in this section to assess Parky's usability and practical value—based on user evaluations through responses to questions and Likert-scale ratings—provides an objective assessment. By combining a structured Likert-scale questionnaire with objective performance metrics, the analysis can demonstrate not only whether users are satisfied with the application but also whether they can successfully and efficiently perform the basic activities of searching for and reserving parking spaces. The data obtained would support the experimental validation of the proposed integrated parking space reservation management system and provide a clear basis for further improvements.

8. Conclusions

The Parky mobile app makes it easier for users to find and book parking spots. It combines the best of the functionalities of well-established similar apps and builds upon them with innovative, user-centric solutions. Its interface is user-friendly, making it convenient to use. Its main advantage is that it is completely free. In addition, it allows users to search for parking by city, displaying available parking spots on a map and providing details for each, such as available spots, usage fees, contact information for the facility, and navigation directions to reach it. It incorporates a variety of useful functions, inspired

by similar apps, as well as new features designed to make it easier for users when arriving in an unknown city. Unfortunately, this first version of the app is available only for parking in Bulgaria. Parky has been adapted to operate in accordance with the specific conditions and legal regulations in Bulgaria, which currently limits its direct implementation in other countries, requiring parking spots outside the country. Compared to similar apps, Parky offers more comprehensive coverage of parking spots in Bulgaria, including the ability to book a parking spot anywhere in the country. It stores recommendations from other users regarding the condition of parking lots and the quality of service, which provides additional information for a more accurate assessment of the quality and available services at different parking spots.

The three-layer Model–View–Presenter architecture used in the application ensures that each part of the system can be developed, improved, and expanded in functionality independently of the others. This ensures greater operational stability and optimization of resource usage.

The proposed parking space reservation system is application-oriented. It does not aim to develop new approaches or techniques to solve the problem but rather implements an existing model (MVP) to be as useful as possible and to make users' daily lives easier by providing them with the necessary information in real time. This allows them to save time for more important professional tasks because "time is money".

Overall, the proposed application contributes by integrating multiple parking management functionalities into a unified cloud-based platform rather than providing an isolated parking service. This functional integration distinguishes Parky from the reviewed applications and provides a flexible foundation for future intelligent parking systems. So, its innovative contribution consists of the development of a fully functional mobile app for reserving parking spots. It is specifically adapted to the unique parking conditions in Bulgaria, and it offers both the useful features of existing similar apps and the new own functions discussed above.

The developed Parky application was evaluated by thirty users according to a set of key criteria, including learnability, navigation, interface clarity, parking-space search, the reservation process, information quality, usefulness of recommendations, reservation management, functional integration, user confidence, intention to use, and overall satisfaction. All criteria were assessed using a standard Likert scale.

At this stage, no direct comparison with similar applications was conducted because the sample of Bulgarian users evaluated only the Parky application.

As part of future work, the evaluation will be extended to a larger sample of participants, comprising more than thirty users. The participants will test both Parky and comparable mobile applications for parking-space reservation under the same experimental conditions. This approach will enable a direct comparison of the evaluation results obtained for the different applications.

The proposed Parky mobile app is designed with the option for future expansion and the addition of new functions. Its structure allows for easy integration of additional components without the need for significant changes to the existing architecture. This presents a challenge related to the scalability of the app's use, which could be a focus for its future development.

In the future, I plan to adapt it to other countries outside Bulgaria (initially within the European Union) to track and reserve available parking spaces, depending on the specific characteristics of each country and the legal regulations in effect there.

Another opportunity for future development is the creation of new versions of Parky—first for iOS and then potentially for desktop. However, the latter may not be necessary, as the app requires a mobile phone for real-time operation.

Funding: This work has been accomplished with financial support by the European Regional Development Fund within the Operational Programme “Bulgarian national recovery and resilience plan”, procedure for direct provision of grants “Establishing of a network of research higher education institutions in Bulgaria”, and under Project BG-RRP-2.004-0005 “Improving the research capacity and quality to achieve international recognition and resilience of TU-Sofia (IDEAS)”.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DB	Database
ITS	Intelligent Transportation System
IoT	Internet of Things
MVP	Model–View–Presenter
RFID	Radio Frequency Identification
XML	Extensible Markup Language

References

1. Foth, M. *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*, IGI Global: Hershey, PA, USA, 2009.
2. Song, H.; Srinivasan, R.; Sookoor, T.; Jeschke, S. *Smart Cities: Foundations, Principles and Applications*; John Wiley & Sons: Hoboken, NJ, USA, 2017.
3. Biyik, C.; Allı, H.; Arslan, A.; Coşkun, E. Smart parking systems: Reviewing the literature, architecture and ways forward. *Smart Cities* **2021**, *4*, 623–642.
4. Douglas, M.; Luke, R.; Twinomurinzi, H.; Mageto, J. From concepts to connectivity: A systematic review of smart mobility antecedents and impacts. *Technol. Forecast. Soc. Change* **2026**, *228*, 124684. <https://doi.org/10.1016/j.techfore.2026.124684>.
5. Kotb, A.O.; Shen, Y.C.; Zhu, X.; Huang, Y. Smart parking guidance, monitoring and reservations: A review. *IEEE Intell. Transp. Syst. Mag.* **2017**, *9*, 6–16.
6. Chowdhury, M.A.; Sadek, A.W. *Fundamentals of Intelligent Transportation Systems Planning*; Artech House: Norwood, MA, USA, 2003.
7. Bahga, A.; Madiseti, V. *Internet of Things: A Hands-on Approach*; Universities Press: Hyderabad, India, 2014.
8. Kamal, R. *Mobile Computing*, 3rd ed.; Oxford University Press: New Delhi, India, 2017.
9. Fahim, A.; Hasan, M.K.; Chowdhury, M.T. Smart parking systems: Comprehensive review based on various aspects. *Heliyon* **2021**, *7*, e07050.
10. Díaz Ogás, M.; Fabregat, R.; Aciar, S.; Juiz, C. A survey of smart parking systems, *Appl. Sci.* **2020**, *10*, 3872.
11. Zhang, X.; Pitera, K.; Wang, Y. Parking reservation techniques: A review of research topics, considerations, and optimization methods. *J. Traffic Transp. Eng. (Engl. Ed.)* **2023**, *10*, 1099–1117. <https://doi.org/10.1016/j.jtte.2023.07.009>.
12. Becerra-Moreno, J.; Hurtado-Beltran, A.; Domínguez-Mota, F.J.; Guerra, A. Evaluation Approaches and Indicator Architectures for Smart Urban Mobility in Smart City Contexts: A Review. *Future Transp.* **2026**, *6*, 113. <https://doi.org/10.3390/future-transp6030113>.
13. Parkopedia Ltd. For Android: Ver. 3.2.0.5. Available online: <https://parkopedia.en.uptodown.com/android> (accessed on 15 January 2025).
14. ParkMe, INRIX, Inc. For Android: Ver. 2.0.49. Available online: https://parkme.en.uptodown.com/android#google_vignette (accessed on 5 June 2026).
15. JustPark Parking. For Android: Ver. 3.170.2. Available online: https://justpark.en.uptodown.com/android#google_vignette (accessed on 2 July 2026).
16. EasyPark AS. For Android: Ver. 3.170.4. Available online: <https://apps.apple.com/us/app/parkme-parking/id417605484> (accessed on 10 July 2026).

17. Syromiatnikov, A.; Weyns, D. A Journey through the Land of Model-View-Design Patterns. In *Proceedings of Working IEEE/IFIP Conference on Software Architecture 2014 (WICSA 2014)*; IEEE: New York, NY, USA, 2014. <https://doi.org/10.1109/WICSA.2014.13>.
18. Cheng, Y.; Domingues, A. O. Advanced Android App Architecture, Real-world app architecture in Kotlin 1.3. In *Android 9, Android Studio 3.2*, 1st ed.; 2026; 404p.
19. Mainkar, P. *Expert Android Programming, Master Skills to Build Enterprise Grade Android Applications*; O'Reilly media: Sebastopol, CA, USA; Packt Publishing: Mumbai, India, 2017.
20. Smyth, N. *Android Studio Otter Essentials—Java Edition: Developing Android Apps Using Android Studio Otter and Java*, 1st ed.; Payload Media: Cary, NC, USA, 2025; 692p.
21. Forrester, A.; Boudjnah, E.; Dumbavan, A.; Tigcal, J. *How to Build Android Applications with Kotlin: A Hands-On Guide to Developing, Testing, and Publishing Production-Grade Android 16 Apps*, 3rd ed.; O'Reilly media: Sebastopol, CA, USA; Packt Publishing: Mumbai, India, 2025; 654p.
22. Darcey, L.; Conder, S. *Android Wireless Application Development*; Addison-Wesley Professional: Reading, MA, USA, 2012.
23. Gookin, D. *Guide to XML and JSON*, 1st ed.; Programming: Boston, MA, USA, 2019; 107p.
24. Mayers, D. S. *Data Structures and Algorithms in Java: A Project-Based Approach*, 1st ed.; Cambridge University Press: Cambridge, UK, 2024.
25. Ashok Kumar, S. *Mastering Firebase for Android Development: Build Real-Time, Scalable, and Cloud-Enabled Android Apps with Firebase*, 1st ed.; O'Reilly media: Sebastopol, CA, USA; Packt Publishing: Mumbai, India, 2018; 394p.
26. Ferrando, P.J.; Morales-Vives, F.; Casas, J.M.; Muñoz, J. *Likert Scales: A Practical Guide to Design, Construction and Use*, 1st ed., Psicothema: Oviedo, Spain, 2025; Volume 37, pp. 1–15. <https://doi.org/10.70478/psicothema.2025.37.24>, 2025.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.