MINISTRY OF EDUCATION AND RESEARCH OF REPUBLIC OF MOLDOVA
TECHNICAL UNIVERSITY OF MOLDOVA
FACULTY OF COMPUTERS, INFORMATICS AND MICROELECTRONICS
DEPARTMENT OF SOFTWARE ENGINEERING AND AUTOMATION

Parking Guru: Application for Parking Reservations
Internship Report

Students:	Bostan Victor, FAF-222
	Dimbițchi Sergiu, FAF-222
	Doschinescu Dan, FAF-222
	Grosu Renat, FAF-222
	Ostafi Eugen, FAF-222
Company Mentors:	Lisa Andrei
	Abner Veaceslav
University Mentor:	Gogoi Elena

Chișinău, 2024

ABSTRACT

The project entiteled ”ParkingGuru” was developed by Bostan Victor, Dimbitchi Sergiu, Doschinescu Dan, Grosu Renat and Ostafi Eugen, enrolled in the Software Engineering program at the Technical University of Moldova. ParkingGuru is a dynamic web application designed to streamline parking management by offering an efficient, user-centric platform for real-time parking reservations. The application allows users to manage reservations, vehicles, and user profiles while incorporating role-based access control and secure authentication through OAuth2 and JWT tokens. Key features include active reservation monitoring, pricing calculations and parking flow management. This report examines the system’s architecture, emphasizing its practical benefits in optimizing urban parking through technology and secure user interaction.

Keywords: parking management, reservation system, user authentication, urban mobility, role-based access control

TABLE OF CONTENTS

INTRODUCTION

In today's rapidly urbanizing world, parking challenges have become a significant issue in cities. This report introduces ParkingGuru, a web application developed to enhance parking management by enabling real-time parking reservations and user-friendly vehicle management. The increasing need for optimized parking solutions and the integration of technology into everyday processes motivated the choice of this topic. ParkingGuru stands out as it addresses the pressing issues of parking efficiency and user convenience, providing a system that simplifies reservations and ensures security through role-based access and authentication mechanisms.

The goal of ParkingGuru is to streamline the process of finding and reserving parking spots, offering users a seamless experience. Additionally, the platform provides a robust backend, featuring reservation management, pricing calculations, and active monitoring of parking sessions. By utilizing modern technologies like JWT for secure authentication and GraphQL for data handling, ParkingGuru blends efficiency and reliability in its design.

The development of ParkingGuru involved research into the current state of parking infrastructure, user interface design, and security protocols, focusing on addressing modern parking challenges. This report outlines the entire development process, discussing both the conceptual and practical aspects, the hurdles faced, and the creative solutions implemented.

ParkingGuru marks a substantial advancement in parking technology, offering a comprehensive solution for urban parking issues and setting a precedent for future developments in this field.

1 DOMAIN ANALYSIS

ParkingGuru is designed to tackle a significant issue in urban settings—parking management. With growing cities and increasing vehicle ownership, parking availability has become a major challenge for drivers, city administrators, and law enforcement. The system seeks to simplify the process of finding, reserving, and paying for parking spaces while also providing tools for monitoring parking compliance. It does so by integrating essential features such as reservation systems for parking spaces, real-time status updates, and fee management, all within a user-friendly mobile application.

The domain analysis of ParkingGuru focuses on solving the problems of parking congestion, inefficient space allocation, and the administrative difficulties of managing parking violations. Traditional parking methods often lead to wasted time, frustrated drivers, and inefficient use of parking facilities. These issues are especially prevalent in busy city centers, where limited space and high demand exacerbate the situation. Furthermore, law enforcement has difficulty monitoring whether parked cars comply with the rules, leading to enforcement inefficiencies and potential revenue losses for city governments.

ParkingGuru is designed for two primary user groups: drivers and law enforcement officers. Drivers need an easy way to find, reserve, and pay for parking spots, while law enforcement officers require a system that allows them to check the status of parking reservations and enforce parking regulations. By addressing the needs of both groups, ParkingGuru creates a more efficient and transparent system for managing parking in urban areas.

This domain analysis provides the foundation for developing a system that improves parking management, reduces congestion, and enhances user experience. By targeting a specific issue and delivering an optimized solution, ParkingGuru aims to offer a practical tool that benefits both users and administrators alike.

1.1 Problem Overview

Parking in urban areas has become a growing issue, particularly as cities expand and the number of vehicles on the road increases. Limited parking spaces, rising vehicle ownership, and urbanization contribute to severe congestion in city centers. This leads to a situation where drivers spend significant time searching for available parking, further exacerbating traffic and increasing pollution levels. Research indicates that drivers spend an average of 17 hours annually searching for parking spots, a statistic that underscores the inefficiencies within current parking systems [**1**].

The lack of real-time parking tracking presents one of the primary challenges for drivers. Without a system to provide updates on available spaces, drivers are left navigating large parking lots or unfamiliar streets without guidance. This increases the likelihood of traffic congestion and driver frustration. Moreover, drivers often forget the exact location of their parked cars, especially in large, crowded lots.

Without the ability to receive real-time updates or map directions, locating a parked vehicle becomes a time-consuming task that reduces overall user satisfaction [**2**].

Another key issue is the absence of dynamic pricing systems. Most current parking management systems rely on static pricing strategies that do not adjust according to demand. For example, during city events, rush hours, or other high-demand periods, the lack of flexible pricing results in inefficient space utilization. Drivers might avoid high-demand areas altogether due to flat-rate pricing, while others pay the same rate even in low-demand times, such as late at night. An intelligent system that adjusts prices based on demand, traffic levels, and special events could improve the overall utilization of available parking spaces [**3**].

Additionally, enforcement of parking rules and monitoring of reservations is challenging for city administrators and law enforcement. Many parking violations go undetected due to the lack of integrated real-time monitoring systems. Law enforcement officers typically rely on manual checks, which are time-consuming and inefficient. With the rapid growth of vehicle ownership, it has become increasingly difficult for officers to monitor every parked car for violations, unpaid reservations, or illegal parking in restricted areas. The absence of a dedicated system for police to track active reservations and violations in real-time limits their ability to efficiently enforce parking rules [**4**].

From a city administrator's perspective, these challenges also result in lost revenue. Unpaid parking fees and uncollected fines due to inefficient monitoring systems lead to potential revenue losses. Moreover, the lack of data-driven insights means that administrators cannot effectively manage and optimize the use of available parking infrastructure.

To address these key problems, ParkingGuru offers a solution that incorporates real-time parking tracking, dynamic pricing based on demand, and dedicated tools for law enforcement officers. By providing these essential features within a user-friendly interface, ParkingGuru aims to improve parking efficiency, reduce traffic congestion, and enhance revenue collection for city governments. Additionally, the platform supports mobile payments, making it easier for drivers to reserve and pay for spaces with minimal friction.

In conclusion, the combination of rising vehicle ownership, outdated parking management systems, and the increasing complexity of urban environments makes it essential for cities to adopt innovative solutions like ParkingGuru. By focusing on the real-time tracking of parking spaces, introducing flexible pricing systems, and equipping law enforcement officers with the necessary tools to monitor violations, ParkingGuru addresses the core challenges of urban parking management. Through this approach, cities can enhance driver experience, optimize the use of parking resources, and increase revenue from parking fees and fines.

1.2 Target Audience

The target audience for the ParkingGuru app consists of several key groups. First, urban drivers who frequently navigate busy city areas and are often frustrated by the lack of available parking spaces. These drivers can benefit from the app's real-time tracking and reservation capabilities, allowing them to avoid long searches for parking spots and reducing the stress of urban driving. The app's dynamic pricing models will also help them save money by offering reduced rates during off-peak hours or in less congested areas.

Next, commuters who drive to work in densely populated city centers are a crucial audience. These individuals often need reliable parking near their offices, and the ParkingGuru app provides the ability to reserve parking spaces in advance. This saves time and helps commuters avoid unexpected parking issues. Additionally, the app's dynamic pricing feature provides flexibility, allowing them to adjust their parking strategy based on their work schedule and proximity to their workplace.

Tourists are another essential audience for ParkingGuru. When visiting new cities, tourists are often unfamiliar with local parking regulations, available spaces, and pricing. The app helps visitors easily locate parking near key landmarks and attractions, providing them with the peace of mind that their vehicle is safely parked. The app's vehicle-tracking feature ensures they can easily find their car after a day of sightseeing, simplifying their experience in a new environment.

City administrators and private parking providers also benefit from ParkingGuru. Managing parking spaces in busy urban centers presents challenges, such as optimizing space usage, tracking revenue, and monitoring parking compliance. With real-time insights, the app enables administrators to efficiently manage parking availability and enforce rules, while also providing a platform for private parking providers to maximize revenue through dynamic pricing models and improved space management.

Finally, law enforcement officers play a crucial role in managing parking violations and enforcing regulations. ParkingGuru offers them a dedicated interface for tracking parking compliance and checking reservations in real time, enabling more effective enforcement. The app allows officers to quickly identify illegal parking and monitor active reservations, streamlining the enforcement process and reducing manual patrols.

By addressing the unique needs of each of these groups, ParkingGuru provides a comprehensive solution for the challenges associated with urban parking, creating a more efficient and transparent parking system that benefits drivers, administrators, and law enforcement alike

1.3 Solution Concept

ParkingGuru offers an advanced and user-friendly solution to parking management in urban settings. It addresses common parking challenges by integrating real-time tracking, dynamic pricing, and law enforcement features into a single mobile platform.

One of its core functionalities is real-time vehicle tracking, which allows users to precisely locate their parked car. This feature is particularly beneficial in large parking lots or unfamiliar areas, helping users avoid the frustration of searching for their vehicles. The system provides a GPS-enabled service that stores the location of the car as soon as the user parks, ensuring that users can easily find their vehicle later.

Another key feature is dynamic pricing, which adjusts parking rates based on factors such as city congestion, peak hours, or special events. This allows users to benefit from lower prices during off-peak times, optimizing parking costs and improving space utilization in high-demand areas. The dynamic pricing model ensures more efficient parking space allocation by encouraging users to park in less congested areas during peak times.

ParkingGuru also includes a law enforcement interface that allows police officers to monitor parking compliance. Through this dedicated interface, officers can check reservation statuses, verify if vehicles have paid for parking, and issue penalties for unauthorized or expired reservations. This reduces manual patrol efforts and enhances the ability to manage parking violations effectively.

In addition, the system removes the need for cash payments by integrating a digital payment system, allowing users to reserve and pay for parking spaces seamlessly through the app. This feature eliminates the hassle of finding cash or dealing with parking meters, streamlining the entire parking process for users.

The reservation system in ParkingGuru allows drivers to reserve parking spots in advance, ensuring they have a guaranteed spot upon arrival. This reduces the time spent searching for parking and provides peace of mind for users, particularly during peak times or in high-demand areas.

By combining these features, ParkingGuru not only simplifies parking management for users but also provides city administrators and law enforcement with effective tools for ensuring compliance, managing parking resources, and reducing congestion. The system improves the overall user experience by providing real-time data, flexible pricing, and seamless payment options, while law enforcement benefits from a streamlined enforcement process.

1.4 Market Research

The market for parking management solutions continues to grow, driven by increasing urbanization, the rise of mobile payments, and the need for more efficient public parking management systems. According to projections, the global smart parking market is expected to reach $15.65 billion by 2027, a sign of the growing demand for innovations in this sector. While existing solutions like ParkMobile and JustPark have primarily focused on private parking management, ParkingGuru is unique in its emphasis on public parking spaces, where the challenge of finding and reserving parking in busy city areas remains largely unmet. ParkingGuru addresses the inefficiencies in urban public parking, where congestion, space management, and compliance monitoring are ongoing issues.

Unlike its competitors, ParkingGuru integrates dynamic pricing strategies, offering drivers the ability to reserve public parking spaces and pay based on real-time demand. This functionality is essential in reducing congestion and making parking more accessible, particularly during peak times or large-scale events. Meanwhile, law enforcement officers benefit from the app's ability to monitor parking compliance through a specialized interface, ensuring that parking violations are minimized, and revenue losses are avoided. The integration of mobile payments, growing at a CAGR of 24.5%, further enhances user convenience, allowing for seamless and contactless transactions.

ParkingGuru's focus on sustainability and smart city development makes it an ideal solution for cities looking to optimize parking and reduce emissions. By decreasing the time drivers spend looking for parking, the app helps lower overall carbon emissions while also contributing to smoother traffic flow in urban centers. Additionally, the app's dynamic pricing based on congestion supports city goals in managing public resources more effectively, aligning with broader initiatives to reduce the environmental impact of growing urban populations.

Finally, market research has shown that the demand for parking apps with integrated law enforcement features is high, and ParkingGuru fills this gap by offering a comprehensive, public-sector-focused solution. The global shift toward smart cities and sustainable urban development further validates the need for ParkingGuru, as cities continue to prioritize tools that help manage their public infrastructure more effectively.

2 SYSTEM DESIGN

The system design phase is a detailed process where we define the software's architecture, components, and interfaces, ensuring alignment with the needs of the target users. This phase is divided into multiple stages, each playing a critical role in shaping the overall design of the application. The results from these stages contribute to the final system structure, which will be further discussed in the subsequent sections of this report.

2.1 Technical Requirements

The next critical step for ParkingGuru is defining and understanding the system's requirements. Requirements analysis is vital to the success of any software product, as it outlines the key attributes and functionalities necessary to meet business goals and user expectations. By clearly identifying both functional and non-functional requirements, ParkingGuru ensures it delivers a seamless and valuable parking experience. In this section, we will explore these requirements and discuss their importance in enhancing the system's overall performance and usability.

2.1.1 Functional Requirements

ParkingGuru is a comprehensive parking reservation system designed to simplify the process of finding and reserving parking spaces for users. The system provides a seamless experience for both users and administrators, ensuring that the essential parking management operations are efficient and user-friendly. Below is a detailed outline of the functional requirements for ParkingGuru:

1. User Registration and Authentication

1.1 User Registration - The system must allow users to register with either an email address or phone number. - A unique phone number, email, and UUID must be used for each registration to ensure that each user is distinct within the system. - Upon registration, the system should generate and send an OTP (One-Time Password) to verify the phone number provided by the user. - Users cannot complete the registration process until they successfully verify their phone number through OTP. - Once registered, the user's profile will contain information such as first name, last name, email, phone number, and a flag indicating whether the account has been verified.

1.2 User Login - Users must be able to log in using either their phone number or email address. - If a phone number is used, the system will authenticate the user by phone number; otherwise, it will authenticate by email. - The login process must return a valid JWT token that will be used for subsequent requests.

1.3 Role-Based Access Control - The system must support two main roles: USER and POLICE. - USER: Regular users of the system who can reserve, and manage parking reservations. - POLICE: Users who have additional access to enforce parking regulations and monitor the system. - The system should restrict access to certain features, such as viewing or managing reservations by plate number, to users with the POLICE role.

2. Reservation Management

2.1 Create Reservation - The system must allow users to create parking reservations by providing their vehicle's plate number and the parking address, which is taken automatically by the frontend service from the geolocation. - Reservations should automatically include a start time (the current time) and should initially set the ‘endDateTime‘ to ‘null‘, indicating that the parking session is active. - Users cannot create a new reservation if they already have an active reservation, i.e., one where ‘endDateTime‘ is ‘null‘. - The system should validate whether the user is allowed to create a reservation and throw an appropriate exception if they attempt to reserve a spot while having an active reservation.

2.2 End Reservation - Users should be able to end their active reservation, at which point the system must record the end time and calculate the total parking fee based on the time elapsed. - The system will store the total price in the reservation record once it is calculated during the end reservation process.

2.3 Active Reservation Retrieval - The system must allow users to retrieve their active reservation. If no active reservation exists, the system should respond with a message indicating that no active reservation is found. - The system should also return the elapsed time since the reservation was created for active reservations.

2.4 Active Reservation by Plate Number (Police Role) - Users with the POLICE role should be able to retrieve active reservations by searching for a specific plate number. - This feature is designed to allow law enforcement or system administrators to monitor and enforce parking regulations.

3. Profile and Account Management

3.1 Profile Management - The system should restrict changing the email or phone number. - Users must also have the ability to view their profile, which will display their personal information.

3.2 Phone Number Verification - The system must allow users to verify their phone numbers via OTP. If the OTP verification is successful, the ‘isVerified‘ flag should be set to ‘true‘. - Phone number verification should be done during the registration process and users should be restricted from creating an account if the phone number was not verified.

These functional requirements detail the essential features that ParkingGuru must deliver. They outline the fundamental operations that users will expect, ensuring the system meets both user and security needs.

2.1.2 Non-Functional Requirements

Non-functional requirements define the criteria that judge the performance, security, usability, and overall quality of ParkingGuru, ensuring it meets user expectations and operational standards. Below is a detailed outline of these requirements:

1. Performance Requirements

1.1 Scalability

The system must be scalable to accommodate an increasing number of users and reservations, ensuring performance does not degrade as the user base grows. ParkingGuru must be capable of handling hundreds of concurrent users making reservations, verifying OTPs, and querying parking availability.

1.2 Response Time

ParkingGuru must respond to user requests within acceptable timeframes: Authentication and OTP verification: Must complete within 2-3 seconds. Creating and ending reservations: Should process within 1-2 seconds. Fetching reservation history or active reservations: Responses should be generated in under 2 seconds.

1.3 Data Consistency

Data related to reservations must remain consistent across the system, ensuring that reservation start and end times, pricing, and status updates are properly synchronized.

2. Security Requirements 2.1 Encryption and Authentication

Data Encryption: All sensitive user data, including personal information and reservation details, must be encrypted both in transit and at rest. Data transmitted over the network must be secured using SSL/TLS protocols, and stored passwords must be hashed using a secure hashing algorithm such as bcrypt. Multi-Factor Authentication (MFA): To enhance user security, the system must implement multi-factor authentication (MFA). Users must verify their identity through an additional OTP (One-Time Password).

2.2 Secure API Design

OAuth 2.0 for Authentication and Authorization: All API endpoints must be protected using OAuth 2.0 for secure authentication and authorization. Access tokens must be issued upon successful login and used to authenticate subsequent API requests. Input Validation and Sanitization: All user inputs must be validated and sanitized to prevent injection attacks, such as SQL injection or command injection. The backend should use parameterized queries or ORM frameworks to ensure that user inputs are handled securely. Secure Error Handling: The system must implement secure error handling to avoid exposing sensitive information. Error messages sent to clients should be generic and not disclose any internal system information (e.g., stack traces or database errors).

2.3 Frontend Security

Cross-Site Scripting (XSS) Protection: All user input fields must be sanitized and encoded to prevent XSS attacks. The frontend must implement a Content Security Policy (CSP) to restrict the execution of untrusted scripts. Cross-Site Request Forgery (CSRF) Prevention: To prevent CSRF attacks, all state-changing requests must be protected with anti-CSRF tokens. This ensures that the system verifies the authenticity of requests before processing them. Secure Cookie Handling: Cookies used for session management or storing sensitive information (such as JWT tokens) must have the following security attributes: HttpOnly:

Prevents cookies from being accessed via JavaScript. Secure: Ensures cookies are only transmitted over HTTPS connections. SameSite: Prevents cookies from being sent in cross-site requests, helping mitigate CSRF attacks.

2.4 Backend Security

SQL Injection Prevention: To protect against SQL injection attacks, the backend must use parameterized queries or an ORM framework like Hibernate or JPA. This ensures that user inputs are safely handled without the risk of executing malicious SQL commands. Session Management: The system must securely manage user sessions. JWT tokens should be used for stateless session management, and sessions must be invalidated after a certain period of inactivity. Tokens should be securely stored and invalidated on logout.

2.5 Role-Based Access Control (RBAC)

User Role Management: The system must implement role-based access control (RBAC) to limit access to specific features and data based on user roles. Users should be classified into roles like USER and POLICE, with each role having distinct permissions. For example, USER accounts should only access their own reservations, while POLICE accounts should have the ability to search for reservations by plate number. Access Restrictions: Unauthorized users attempting to access restricted features should be denied with appropriate error messages. Attempted violations must be logged for further investigation.

3. Usability Requirements 3.1 User-Friendly Interface

ParkingGuru's UI must be intuitive and easy to navigate. Features such as registration, reservation creation, and profile management should require minimal training. The design must follow a mobile-first approach to ensure that the application is accessible on both mobile and desktop devices.

3.2 Error Handling and Feedback

Users should receive clear and actionable error messages. If a user is unable to make a reservation because they already have an active one, the system should provide detailed feedback with corrective steps.

These non-functional requirements define the operational standards for ParkingGuru. They ensure the system meets performance, security, usability, and reliability benchmarks, providing a robust and user-friendly solution for parking management.

2.2 Behavioral Modeling

Behavioral modeling focuses on representing the dynamic aspects of a system, specifically how it behaves in response to various inputs or events. Use Case Diagrams depict the interactions between actors (users or other systems) and the system, outlining the different ways the system can be used to fulfill specific goals. Sequence Diagrams illustrate the chronological order of messages and interactions between objects or components, showing how processes are executed within the system. Behavioral modeling provides insight into the functional flow and processes that occur within the system, focusing on its dynamic nature.

2.2.1 Use Case Diagrams

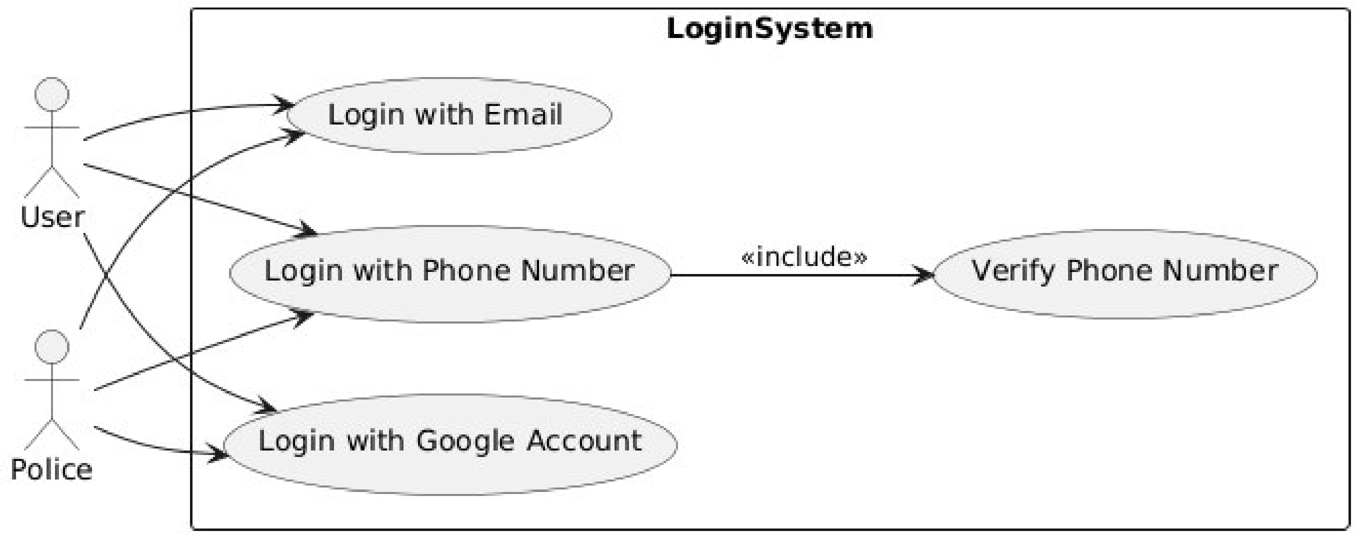

Figure 2.1 – Use Case Diagram for Login System

The figure 2.1 illustrates the ”Login System” functionality within the ParkingGuru application. Two types of users, ”User” and ”Police,” have the ability to log in using different methods. Both actors can log in with either their email, phone number, or Google account.

The ”Login with Email” and ”Login with Google Account” use cases provide straightforward login methods for users. In contrast, the ”Login with Phone Number” use case involves an additional verification step, represented by the ”Verify Phone Number” use case, which includes the phone login process. This indicates that verifying the phone number is an essential part of phone-based login, ensuring that users can only proceed after validating their identity via a code sent to their phone.

The diagram captures the flexibility and security of the login process, accommodating different user preferences and ensuring identity verification, especially for phone-based login scenarios.

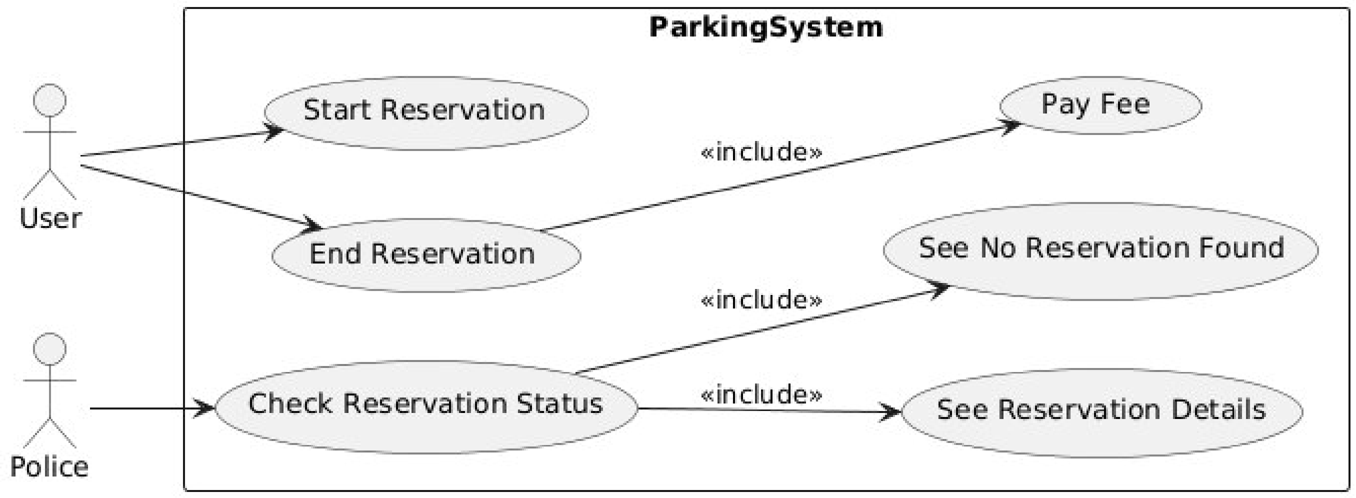

Figure 2.2 – Use Case Diagram for Parking System

The figure 2.2 illustrates the ”Parking System” functionality within the ParkingGuru application. Two actors are involved: the ”User” and the ”Police.” Each actor interacts with the system in different ways.

The ”User” can start and end a reservation. When a user ends a reservation, they must also pay the associated parking fee, which is represented by the ”Pay Fee” use case, included in the ”End Reservation” process. This ensures that the user cannot complete the reservation process without handling payment.

The ”Police” actor is responsible for checking the status of reservations. When the police check the reservation status, they are either presented with reservation details or notified that no reservation has been found. These two actions are included in the ”Check Reservation Status” process, as the outcome depends on whether the given car has an active reservation or not.

The use of ”include” relationships shows that certain actions (such as paying the fee or viewing reservation details) are integral steps within the primary operations of ending a reservation or checking the reservation status. This diagram highlights the central features of the system, from parking management for users to enforcement and validation by police.

2.2.2 Sequence Diagrams

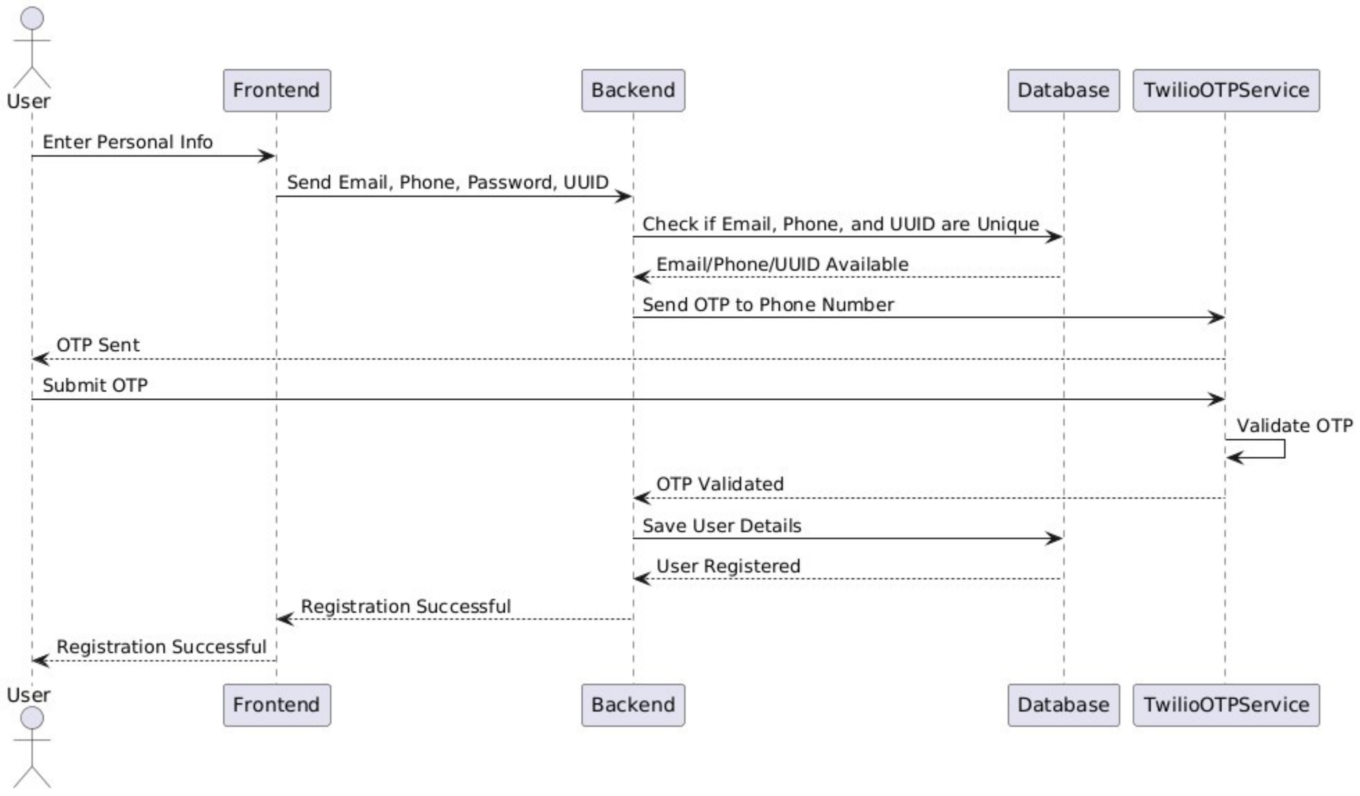

Figure 2.3 – Sequence Diagram for Registration

The sequence diagram in the figure 2.3 illustrates the registration process for the ParkingGuru application. The interaction begins with the user entering their personal information, including their email, phone number, password, and UUID. This information is sent to the backend for validation. The backend checks if the provided email, phone number, and UUID are unique by querying the database. If they are available, the backend proceeds to send an OTP (One-Time Password) to the user's phone number via the Twilio OTP service.

Upon receiving the OTP, the user submits it to the system. The backend, in turn, validates the OTP by communicating with the TwilioOTPService. Once the OTP is successfully validated, the backend saves the user's details to the database, registering the user into the system. The backend then notifies the frontend that the registration is successful, and the user is informed that their registration process has been completed.

This diagram captures both the user's interaction with the frontend and the backend processes that handle validation, OTP generation, and registration, ensuring that the system securely registers users by verifying their phone number.

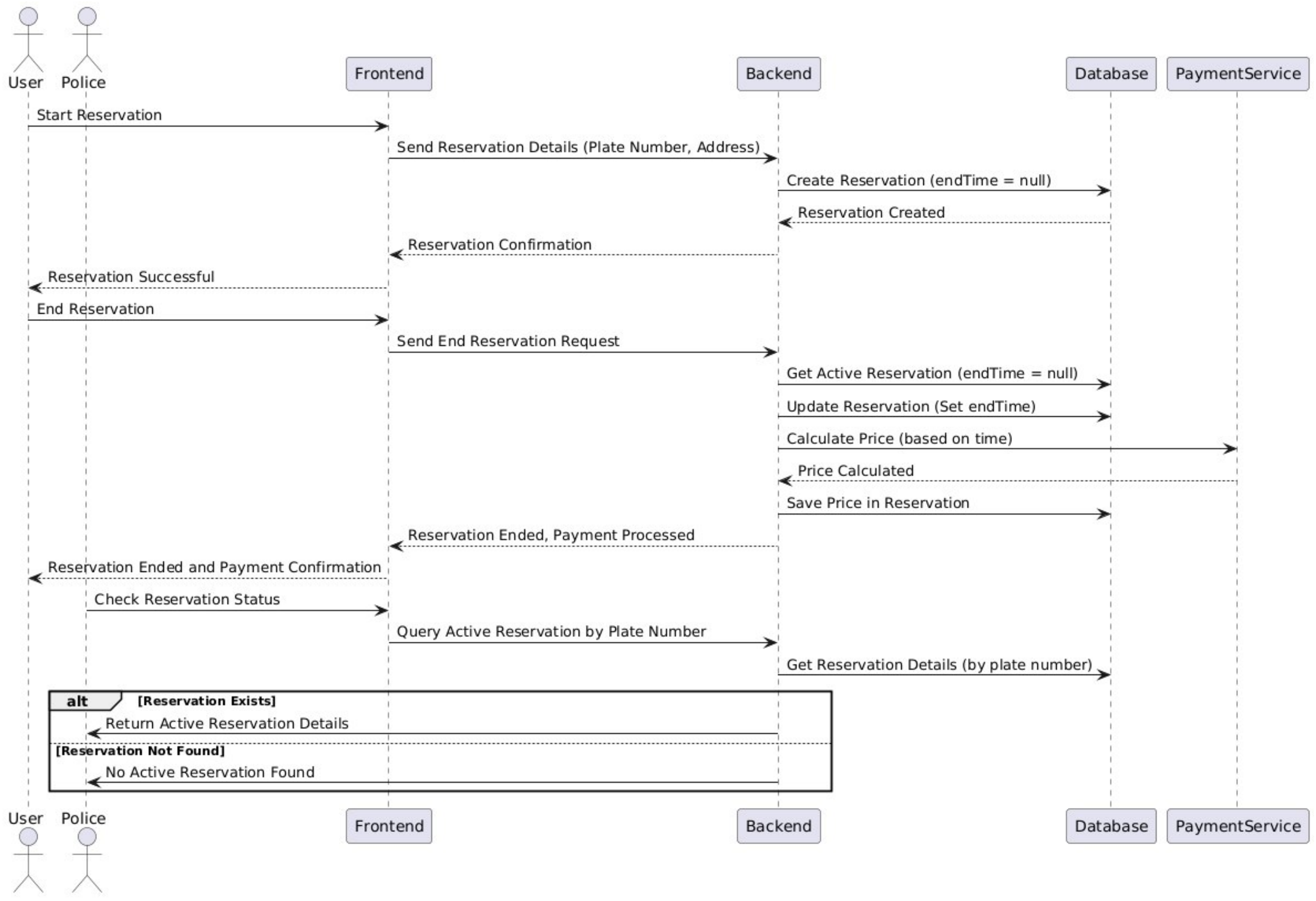
User
Police
Frontend
Backend
Database
PaymentService
Start Reservation
Send Reservation Details (Plate Number, Address)
Create Reservation (endTime = null)
Reservation Created
Reservation Confirmation
Reservation Successful
End Reservation
Send End Reservation Request
Get Active Reservation (endTime = null)
Update Reservation (Set endTime)
Calculate Price (based on time)
Price Calculated
Save Price in Reservation
Reservation Ended, Payment Processed
Reservation Ended and Payment Confirmation
Check Reservation Status
Query Active Reservation by Plate Number
Get Reservation Details (by plate number)
alt
[Reservation Exists]
Return Active Reservation Details
[Reservation Not Found]
No Active Reservation Found

Figure 2.4 – Sequence Diagram for Reservations

The sequence diagram in the figure 2.4 illustrates the reservation system workflow in the Parking-Guru application. The interaction begins with the user starting a reservation by providing the plate number and address. The frontend sends this reservation data to the backend, where a new reservation is created with an empty endTime (indicating it is an active reservation). The reservation is confirmed and stored in the database, after which the frontend receives a reservation confirmation message.

When the user decides to end the reservation, they send a request to the system. The backend retrieves the active reservation (where endTime is null) and updates the endTime to mark the reservation as completed. At this point, the backend calculates the total parking fee based on the time elapsed between the start and end of the reservation. This fee is calculated and stored in the database. The system confirms that the reservation has ended and the payment process has been handled successfully, with the payment details managed by the payment service.

Additionally, the police actor can check the reservation status by querying the system for an active reservation using a specific plate number. If an active reservation exists, the system returns the reservation details; if not, the system responds that no active reservation was found.

This diagram showcases how both users and police interact with the system to manage and check parking reservations, emphasizing the flow of creating, ending, and querying reservations.

2.3 Structural Modeling

On the other hand, Structural Modeling represents the static architecture of the system, focusing on the internal organization of its components. Class Diagrams are commonly used to model the relationships between classes, their attributes, methods, and associations. This provides a blueprint of the system's components and their relationships. Component Diagrams depict the high-level structure, showcasing how different modules or components within the system interact and depend on each other. Structural modeling gives a detailed view of the system's architecture, illustrating the system's organization from a technical perspective.

2.3.1 Class Diagrams

The class diagram represented in the figure 2.1 illustrates the structure of the authentication system in the ParkingGuru application. It consists of the User class, which represents the user of the system. The User class contains attributes such as id, firstName, lastName, email, phoneNumber, password, uid, isVerified, and role. The role of the user is represented by the Role enumeration, which can either be USER or POLICE. The User class also has a method authenticate() to handle user authentication.

The system supports different authentication methods, which are represented by the AuthMethod interface. The interface includes a login() method that must be implemented by the specific authentication methods. Three concrete classes implement this interface:

EmailAuth – This class handles authentication using an email. It contains an email attribute and implements the login() method for email-based authentication.

PhoneAuth – This class manages authentication through a phone number. It contains a phoneNumber attribute and implements both the verifyOTP() method for OTP (One-Time Password) verification and the login() method for phone-based authentication.

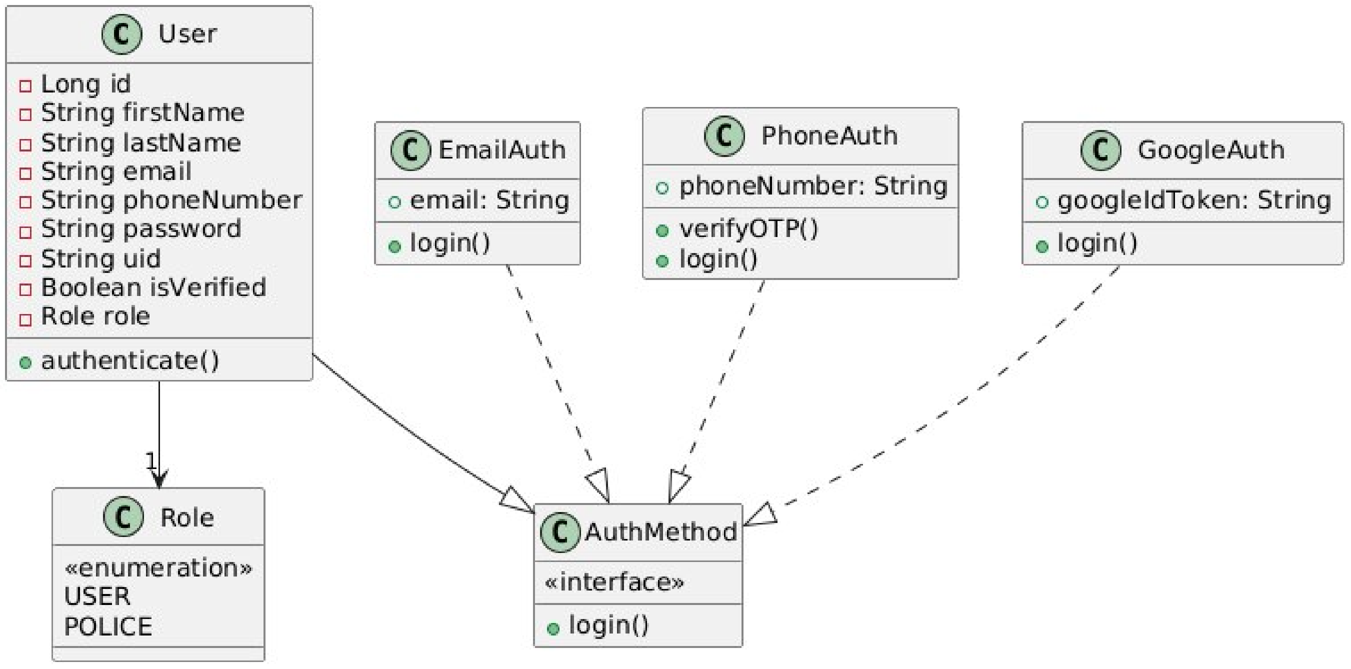

Figure 2.1 – Class Diagram for Login

GoogleAuth – This class provides authentication using Google, with a googleIdToken attribute and an implementation of the login() method for Google account-based authentication.

The diagram shows that the User class interacts with different authentication methods, enabling users to log in via email, phone number (with OTP verification), or Google accounts. The use of the AuthMethod interface ensures that the system is flexible and can easily support additional authentication methods in the future.

The class diagram illustrated in the figure 2.2 the core entities and interactions in the ParkingGuru reservation system.

The User class represents the system's users, with attributes such as id, email, and role. It includes two methods: createReservation() to start a parking reservation and endReservation() to conclude the reservation.

The Reservation class represents a parking reservation, associated with a User. It includes attributes such as startDateTime, endDateTime, plateNumber, address, totalPrice, and a status field of type ReservationStatus. The class also contains a calculatePrice() method, which determines the total price based on the duration of the reservation.

The ReservationStatus is an enumeration that defines the various statuses a reservation can have: UNCHECKED, CHECKED, TICKETED, and NO$_{T}ICKET$.

The diagram shows the relationship between the User and Reservation classes, where one User can create multiple Reservations, and each Reservation has a status indicating its current state. This structure allows the system to manage reservations and track their status, ensuring accurate pricing and compliance

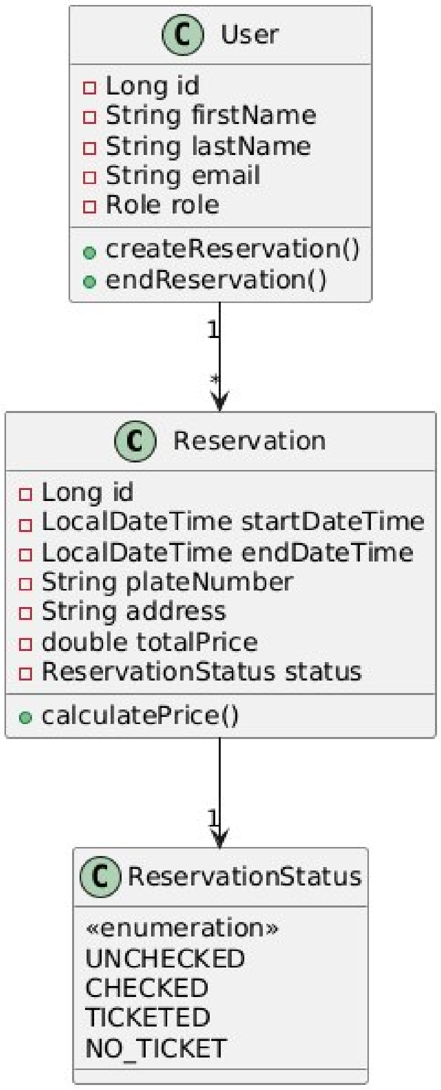

Figure 2.2 – Class Diagram for Reservations

with parking rules.

CONCLUSIONS

In conclusion, the ParkingGuru project has navigated a robust development journey, evolving from a conceptual idea into a fully functional urban parking management platform. Each phase, from domain analysis to the final system design, has contributed to creating an innovative solution for public parking management in congested urban areas.

The domain analysis identified the critical challenges drivers and city officials face, including inefficient space allocation, manual enforcement processes, and growing urban congestion. By addressing these pain points, ParkingGuru has become a streamlined solution designed to enhance both user convenience and administrative control.

Our solution concept, which integrates parking reservations, real-time tracking, dynamic pricing, and law enforcement tools, was validated through targeted market research. This validation demonstrated ParkingGuru's competitive advantage over private parking systems by focusing on public infrastructure, thus meeting the needs of growing urban populations and city governments.

The system design process translated functional and non-functional requirements into actionable plans. Leveraging structural and behavioral UML diagrams, we ensured that both drivers and law enforcement officers experience seamless interaction with the system. Furthermore, the integration of scalable backend services, dynamic pricing, and mobile payments ensures that the system is both robust and user-friendly.

Throughout the agile development process, the implementation of strict security measures, including OAuth 2.0 and secure data encryption, safeguarded user data and protected transactions. The use of GitHub for version control and Docker for deployment streamlined our development workflow, ensuring an optimized system.

In deploying ParkingGuru, we overcame challenges in integrating real-time tracking, law enforcement interfaces, and mobile payments. Our efforts led to a scalable and reliable platform capable of adapting to the evolving needs of urban parking management.

The ParkingGuru project exemplifies our commitment to quality, scalability, and user-focused design. Moving forward, ParkingGuru is well-positioned to revolutionize public parking management in cities worldwide, ensuring that it continues to innovate and address the parking challenges faced by modern cities.

BIBLIOGRAPHY

[1] Kevin McCoy ”Drivers spend an average of 17 hours a year searching for parking spots” Last accessed 16.01.23 https://www.usatoday.com/story/money/2017/07/12/parking-pain-causes-financial-and-personal-strain/467637001/

[2] Patrick Sisson ”Cities have a parking problem. More parking is not the solution.” Last accessed 16.01.23 https://citymonitor.ai/transport/cities-have-a-parking-problem-more-parking-is-not-the-solution

[3] Brad Plumber ”Cars take up way too much space in cities. New technology could change that.” Last accessed 16.01.23 https://www.vox.com/a/new-economy-future/cars-cities-technologies

[4] Eric Brandt ”New Study Shows How Much Time and Money We Waste on Parking” Last accessed 16.01.23 https://www.thedrive.com/article/12389/new-study-shows-how-much-time-and-money-we-waste-on-parking