INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Smart Parking System with Dynamic Pricing Model

G Raghuraman[1]*, S Kavitha[1]*, Nitish Gokulakrishnan[2], Santosh Radhakrishnan[2], Pranav Ramakrishnan[2]

1 Associate Professor, Sri Sivasubramaniya Nadar College of Engineering, Chennai, Tamil Nadu, India
2 UG Student, Sri Sivasubramaniya Nadar College of Engineering, Chennai, Tamil Nadu, India

OPEN ACCESS

Received: 27-12-2022

Accepted: 30-03-2023

Published: 05-05-2023

Citation: Raghuraman G, Kavitha S, Gokulakrishnan N, Radhakrishnan S, Ramakrishnan P (2023) Smart Parking System with Dynamic Pricing Model. Indian Journal of Science and Technology 16(18): 1332-1339. https://doi.org/10.17485/IJST/v16i18.2489

* **Corresponding authors**.

raghuramang@ssn.edu.in

kavithas@ssn.edu.in

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: A Smart Parking System with Dynamic Pricing Model is proposed and developed for vehicle parking management using real time data collected from five popular hotspots in the city of Chennai. **Methods**: The proposed model is developed using polynomial regression algorithm in Python language. The vital parameters considered for training a model are: traffic data, geological distance, time of day, peak hour or not, weekend or not and especially, the number of slots available in the parking lot at the time the user wishes to book a slot in that location. Using these parameters, the model predicts an approximate price for occupying the selected parking slot. For ease of use the dynamic pricing model for parking slot is developed as a GUI using NodeJS for end users. **Findings**: The comparison between static (actual) price with dynamic pricing is done and from the results it is observed that the predicted price is significantly close the actual price. In addition, the dynamic pricing and the availability parking slot can be checked ahead of time for different time slots (normal, peak) to save the time and amount. **Novelty**: (i). Dataset collection — In which data are collected from five different locations of Chennai for both weekdays and weekends, peak time and normal time, approximately 10000 samples. (ii). Proposed techniques — Different technologies (ReactJS, MongoDB, GPS) and linear polynomial regression technique for dynamic pricing is applied for model creation using the values collected from existing live locations. (iii). Parking slot can be booked ahead of time based on user preference using GPS and the payment can be done when physically occupying the slot, the booked slot is reflected in the database and overall count is reduced by one. In overall, the proposed system reduces user productive time, fuel consumption and waiting time.

Keywords: Dynamic pricing; Polynomial regression; Prediction; Parking slot; Time of day; Occupancy

1 Introduction

In current era, the densely populated metropolitan cities are finding difficulty for parking spaces and their respective prices has always been a hassle owing to the variable influx of traffic in different parts of the city[1]. This leads to many drawbacks in real time such as people are moving to sparse locations, traveling on roads with high vehicle density and a worst-case scenario is concerned living without parking space, loss of time and effort[2]. All these drawbacks, affects the people and environment either directly or indirectly.

As a follow up, the smart parking system approach for real time environment is getting more popular to manage and use the existing parking slots effectively. Many researchers are working to develop an efficient parking system for real time use cases for different cities.

In the review paper of Abrar et al., 2021, the comparative analysis of smart parking systems in terms of sensors used, networking technologies, computational approaches, and types of service provided are discussed[3]. Kaari et al., 2022 and Sarker et al., 2020, collected the samples through sensors such as Passive InfraRed (PIR) sensor and Microcontroller based sensor[4,5]. Kaari et al., 2022, represented the problem as a game-theory-based pricing strategy to obtain the best prices using dynamic car pricing system and also to control traffic congestion near parking lots[4]. Sarker et al., 2020, collected the samples through sensors placed underground or ceiling in a multistorey car parking to detect parked vehicles and the price is calculated using edge-cloud computation[5].

At the next level, IoT is used in allocating the parking slots and its equivalent price is calculated. Thanh Nam Pham et al., 2015 through their research, explore the applications of the cloud in the domain of smart-parking[6]. They introduce a novel algorithm that improves the efficiency of the current cloud based smart-parking system and develops a network architecture based on the Internet-of-things technology. The simulation results of their final application show that their algorithm aids in improving the rate of successfully parking the vehicle while minimizing the user waiting time. Poh et al., 2023, collected the dataset from two locations (Kuala Lumpur - 165 slots and Malaysia - 950 slots) for 6 months and deep reinforcement learning with Markov decision process is used in dynamic pricing approach for price prediction[7].

Bibi et al., 2017 aims to optimize the identification of available parking slots in a bid to potentially reduce the bottleneck at a particular parking space/lot[8]. The system proposed involves, taking live stream video footage of vehicles at the parking lot, and providing it as input, from which the number of free and reserved parking slots is obtained through digital image processing techniques. Amir O. Kotb et al., 2016, proposed a smart car-parking system based on dynamic resource allocation and pricing[9]. With a similar end goal in mind, that being to alleviate the issue of parking space dearth due to traffic density, they propose to solve said problems by basing their system on mixed-integer linear programming (MILP) with the objective of reducing the total cost of parking for drivers. The caveats with this however were the scalability, or lack thereof, given that their research, for now only focuses on indoor navigation services for car parking.

However, Mondal et al., 2021 and Shangbin et al., 2022, collected and used the real time dataset in smart parking using database management and circular allocation methodology[10,11]. But the number of constraints and allocation approach are narrow down to the specific region, lacks in scalability.

The limitations of Existing systems are, parking related dataset used is limited to samples collected for two months, single parking region, allocation is carried out on the specific parking location using IoT devices with suitable techniques. The proposed model addressed many of the limitations such as dataset used for training is collected for one year duration with five different popular locations in “Chennai” and the slot booking can be done ahead of time from home itself, which reduces the unnecessary waiting time, anxiety towards availability of parking slot, and saves the fuel too. The new approach adopted is that the required parking slot can be booked with time using GPS enabled approach from any place of the city. In addition, ReactJS is used for creating an user interface and Mongo DB for data management, replaces the traditional, static pricing approach which has no fairness and cost effectiveness in real world environment.

The remaining sections are: In Section 2, the existing approaches for smart parking and the importance of dynamic pricing are explained, the proposed design and methodology are explained with algorithm in Section 3, the results obtained are compared and analyzed using the real time data collected from five parking slots in “Chennai” in Section 4, finally the inferences and future scope are summarized in conclusion.

2 Methodology

The overall workflow diagram of the proposed system on ‘smart parking system with dynamic pricing’ is shown in Figure 1. The user opens the application and enters his desired destination. The current location is detected using the application and a request for all available parking spaces in and around the desired destination is displayed. The price of parking the car in a particular parking slot is generated by the pricing engine and the user can thereafter proceed to book a slot. The selected slot is then allocated for the user and the same is reflected in the database where the number of available slots in that parking lot is reduced by one. The parking lot database contains the following data:

- Time at which the vehicle has entered.
- Location of the parking lot(coordinates)
- Hours for which the slot has been occupied.
- Live number of spaces available in that particular parking lot.

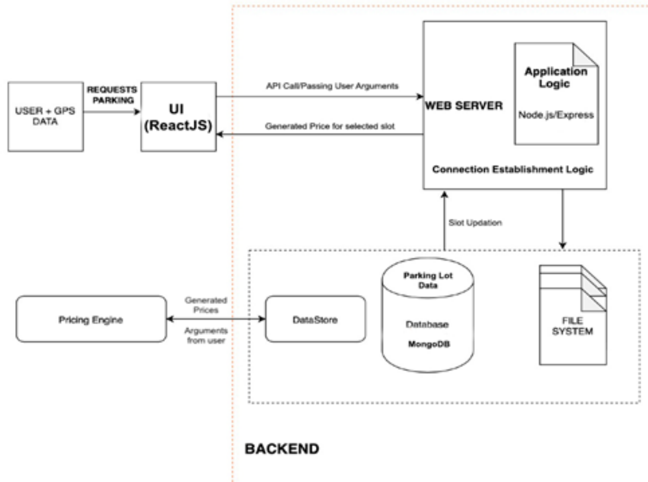

Fig 1. Workflow diagram of 'smart parking system with dynamic pricing' with UI and Backend process

The file system contains the required and relevant HTML and CSS files. The data store contains the values generated by the pricing engine and it is collected Via user interface (ReactJs)[(12,13)]. The details of the parking lots and spaces given by the parking slot providers are to be stored in the Mongo database[(14)]. The application asks the user to enter his current location for two reasons, the first is that the application uses the distance between the current location and the destination as one metric to determine the price to park in that slot i.e. if the user must travel a longer distance to park in that particular spot, the price will be reduced. In addition, the current location[(15,16)] can be used in cases where the user may not book the parking slot in advance, however, chooses to open the application at the time he needs to park. At this time, the application will display the parking spaces that are available nearby so that he may book a slot. Once the user has left the parking space, the number of slots available will also be updated in the database.

2.1 Dataset Description

The dataset collected from the parking slots of five popular hotspots in the city of "Chennai" is used to train and build the dynamic pricing prediction model of the proposed system. The real time parking data of five popular hotspots in the city of "Chennai" are as given below.

- Express Avenue Mall, Royapettah
- Phoenix Market city Mall, Velachery
- Tidel Park IT Sector, Madhya Kailash
- Mayajaal Multiplex, ECR
- Ampa Skywalk Mall, Anna Nagar

The dataset consists of 10,193 samples of data in total. A common feature among all the parking locations is that the earliest possible time of entry is 8:00 AM and the latest is 11:59 PM. The data contains the values recorded between the years 2018 and 2019. The initial dataset had a lot of inconsistencies and after pre-processing, the final dataset was derived successfully.

2.2 Dataset Parameters

The parameters chosen after pre-processing are listed below and it is used for creating a regression model.

- PARKING LOT: This column contains the name of the location of the parking lot.

- COORDINATES: The coordinates are provided as two columns, latitude and longitude and these are the exact coordinates of the parking lots.
- ENTRY TIME: This column describes the exact time at which the vehicle has entered and been parked. This value is imperative to calculate the number of hours that the vehicle has been parked in a particular slot for.
- DURATION: This column describes how long a vehicle has been parked in a lot for as it is one of the primary metrics used to determine the final price that must be paid for parking the vehicle in that slot. The metric for duration in this dataset is hours.
- NUMBER OF AVAILABLE SLOTS: This column contains perhaps the most important information required and it is the number of slots that is available at a particular parking lot at the time the vehicle enters the lot. From the dataset, it can be concluded logically that there are a greater number of available parking spaces during the day as opposed to the evening or night-time.
- AM/PM: This column simply describes whether the vehicle has been parked at the slot before or after noon.
- DATE: This column describes the date on which the vehicle was parked at a slot in the parking lot.
- WEEKEND: This column is imperative as it describes whether the day the car was parked in a slot at the parking lot was a weekend or not. This is described as a Boolean value, TRUE implying it was a weekend and FALSE implying that it was a weekday. This is also an important metric to determine the price of the parking slot as generally, the prices are higher on the weekends due to higher demand and as a result, fewer available parking spaces at a given time.
- PEAK OR NOT: This column describes whether the time at which the vehicle enters to park in a slot is a peak hour or not. The term "peak "implies that there is high demand for parking slots at that time and that there are far less parking slots during these hours. This is an incredibly important metric for our price prediction engine as, using this, we can generate lower prices during non-peak hours and relatively higher prices during peak hours. From the data it can be concluded that generally, evening/nighttime and weekends tend to be the peak hours for the parking lots.
- RATE PER HOUR: This column describes the rate that is charged for parking in a particular slot per hour. These prices are static and that is what our algorithm aims to alleviate. The prices in the dataset are at the discretion of the parking provider.
- TOTAL RATE: This column describes the total amount that the customer must pay for parking their vehicle in a parking slot. This is determined by considering the rate of occupying that particular slot per hour and the number of hours for which the slot has been occupied by the vehicle.

2.3 Polynomial Regression Algorithm

The proposed 'dynamic price prediction model' is developed using polynomial regression algorithm in Python language. The metrics (parameters) used to generate a dynamic price include, time of day, whether it is AM or PM, whether or not it is peak hour, whether it is a weekend or not and especially, the number of slots available in the parking lot at the time the user wishes to book a slot. In addition, the rate will be multiplied based on the number of hours that the user chooses to park in the specific slot.

Algorithm: Price Prediction
Output: Price for the available parking slots
Initialization: Read the 'n' data records
- Encode the data using Label Encoder
- Add Polynomial Features (degree=4) to the training data set using sklearn library
- Import Linear Regression () from sklearn library.
- Fit the polynomial training data and the corresponding output vector containing prices to the Linear Regression model
- Call the predict method of the Linear Regression model.
- Input the predicted prices to the distance_metric() function to obtain the prices after considering the distance metric factor.

We experimented with many other regression algorithms such as multiple linear regression, Random Forest regression, Decision tree regression, Ridge regression and XGBoost regression and concluded that Polynomial regression was the most efficient and provided results with the highest accuracy and lowest standard deviation.

There were a few challenges since the dataset contained a lot of textual data and not much numerical data. The parameters with textual data are, whether or not it is a weekend, whether or not it is a peak hour and the parking slot name. As a result, the textual data had to be encoded to make it usable to the prediction model. Label Encoder and OneHotEncoder classes from sklearn library were used to obtain an encoded data, are used to predict a price. The arguments that the algorithm uses to predict the price are received from the front end, when the user selects the "book" option. It takes in the number of available spaces from the database, depending on the parking lot at which the user has decided to park the vehicle. It extracts whether it is a weekend, and whether it is a peak hour (trained from training dataset). These arguments are passed to the algorithm using NodeJS as the middleware and once the price prediction algorithm is applied on the variables, the price is generated. The generated price is then passed back to the frontend again, where NodeJS is acting as the middleware. The results that were obtained using different cases are given in the following section.

3 Results and Discussion

This section serves as an archive of the results obtained by executing the application under various circumstances (cases) such as allocating the parking slot from the available lots during regular versus peak hours, comparison of actual versus predicted price through dynamic pricing model and performance analysis of the algorithm versus application.

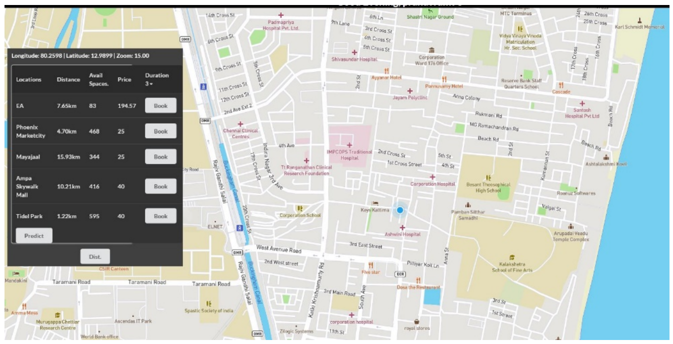

Fig 2. Prices generated by pricing engine during peak hours in EA

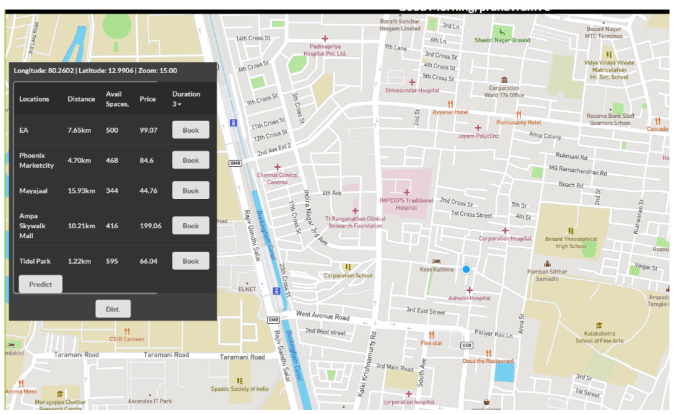

Fig 3. Prices generated by pricing engine during regular hours in EA

The Figure 2 represents the situation where, a vehicle is to be parked in a slot in Express Avenue (EA), for a duration of 3 hours, when there are less than 100 remaining available slots in that parking lot, on a weekend when it is in fact a peak hour, the price generated by the pricing engine is 194.57 rupees. However, it is clear from Figure 3, that when the same user from the same location chooses to park his vehicle at Express Avenue (EA) for the same duration (time), on a Weekday at 9 AM, the price generated by the pricing engine is drastically lower than the peak hours as 99.07 rupees only.

The various cases of pricing calculation obtained by the application for a parking slot are analyzed as follows

- Comparison between the static prices (actual) versus dynamic prices (predicted) for the same destination as given in Table 1.
- Comparison of the prices generated from different locations for the same destination.
- Comparison of the prices from same location and destination, however at different times of the day.

Table 1. Static prices (actual) versus dynamic prices (predicted) from Thiruvanmiyur

Location	Distance (in KM's)	Available spaces	Static price (actual)	Dynamic price (predicted)
Express Avenue (EA)	7.65	200	75	49.33
Phoenix Market City	4.70	200	100	37.79
Mayajaal	15.93	200	60	25
Ampa Skywalk	10.21	200	70	53.44
Tidal Park	1.22	200	35	32.88

In Table 1, the actual prices and the predicted prices generated by the application for parking at five different locations from the same location (Thiruvanmiyur) is given with required details. From the results, we observed the distinction between static and dynamic pricing and the advantage of dynamic pricing provides the user in terms of cost-effectiveness is illustrated.

Table 2. The prices generated by the pricing engine with user being present at Royapettah

Location	Distance (in KM's)	Available spaces	Dynamic price (predicted)
Express Avenue	0.59	200	60.71
Phoenix Market City	8.67	200	49.12
Mayajaal	23.01	200	25
Ampa Skywalk	5.29	200	43.42
Tidal Park	7.31	200	26.65

In Table 2 and Table 3, the prices generated by considering distance as a metric from different locations such as "Royapettah" and "Anna Nagar" are tabulated for the same date and time. Also, in Table 1, as discussed earlier, the prices generated when the user's location is "Thiruvanmiyur" is also considered. From these data, it is clear that even with one distinguishing metric (distance), the prices vary due to the inverse co-relation between the destination and the user's current location.

Table 3. The prices for parking at 10AM with user being present at Anna Nagar

Location	Distance (in KM's)	Available spaces	Price during peak time (10 PM)	Price during normal time (10 AM)
EA	6.73	200	99.17	32.5
Phoenix Market City	10.47	200	120.57	40
Mayajaal	26.62	200	25	39.81
Ampa Skywalk	1.90	200	74.93	25
Tidal Park	11.50	200	36.75	25

The Table 3, on the other hand, represents the prices generated for the user where the location is the same however times are entirely different. Table 3 consists of the prices generated at 10AM and 10PM on the same day. With this, it is concluded that the prices generated differs with respect to the time of day.

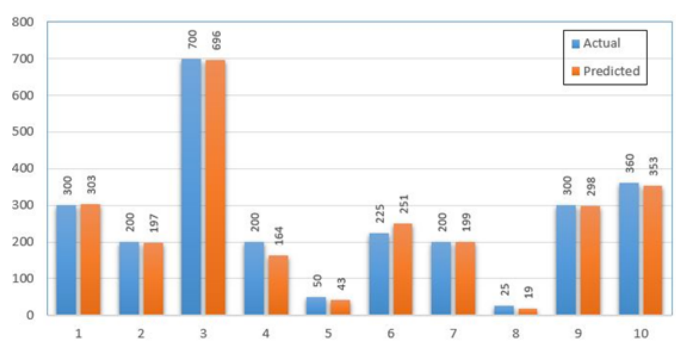

Fig 4. Performance Evaluation: Actual vs Predicted Prices

In addition,Figure 4 and Figure 5 shows the performance of the application with respect to accuracy. The comparison between the actual and predicted prices for a given circumstance is represented. The results imply that the predicted price values are extremely close to the actual price values.

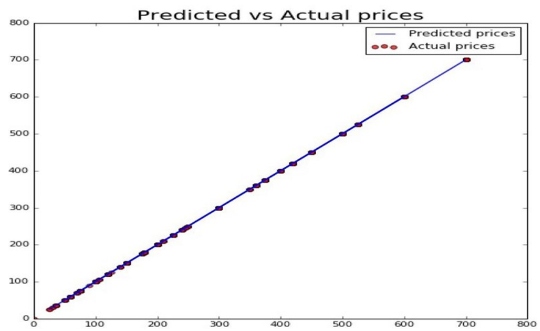

Fig 5. Performance Evaluation: Actual vs Predicted Prices

4 Conclusion

The dynamic price of the parking slot is predicted by considering the distance metric, time of the day and available parking slots using polynomial regression algorithm. From the results it is observed that the predicted price is significantly less than the actual price for both normal and peak hours. For ease of use the dynamic pricing module for parking slot is developed as a GUI using NodeJS for end users. Currently the proposed system is trained using polynomial regression algorithm and tested with five parking locations of "Chennai" from any place of city. From the results it is observed that the predicted prices are significantly closer to the actual price for normal hour. However, a variation occurs during peak hours, where the proposed system recommends minimum price than the actual price at real time locations. During the peak time from Anna Nagar to different locations of "Chennai" city, the average cost is increased from Rs. 12/- to Rs. 81/- when compared to normal time. Also, the parking cost during the peak time varies depends on the starting points. It is inferred that fare varies by Rs. 6 and Rs. 11 from "Thiruvanmiyur" and "Royapettah" respectively and the starting point is computed using GPS location. By considering all these factors, the pre-booking process for parking slot reduces the bottleneck of last-minute tension, cost effective and reduces the fuel consumption. Also, it is possible to scale this research work by adding more locations and its corresponding training data to the existing modules.

From the results it is observed that the predicted prices are significantly closer to the actual price for normal hour. However, a variation occurs during peak hours, where the proposed system recommends minimum price than the actual price at real time locations. During the peak time from Anna Nagar to different locations of "Chennai" city, the average cost is increased from Rs. 12/- to Rs. 81/- when compared to normal time. Also, the parking cost during the peak time varies depends on the starting points. It is inferred that fare varies by Rs. 6 and Rs. 11 from "Thiruvanmiyur" and "Royapettah" respectively and the starting point is computed using GPS location. By considering all these factors, the pre-booking process for parking slot reduces the bottleneck of last-minute tension, cost effective and reduces the fuel consumption. Also, it is possible to scale this research work by adding more locations and its corresponding training data to the existing modules.

In future work, the entire research can be considered as a single module in the grand scheme of things using IoT devices administered at the actual parking location in order to detect whether or not a vehicle has been parked. Also, a payment gateway is required for the user to pay once he has vacated the slot. There is extremely large scope for this idea, and we will continue to build and release as a mobile application for the users with the required improvement. This is a field whose latent potential has not been realized and tapped into until very recently, and we hope to have a production ready full-fledged application at the earliest.

References

1) Gatha S, Pratiksha D, Pranita K, Sneha I, Rudrawar S. Automatic Car parking system using Google Assistant. In: and others, editor. International Conference on Communication information and Computing Technology (ICCICT). IEEE. 2021. Available from: https://doi.org/10.1109/ICCICT50803.2021.9510139.
2) Dutta A, Bhattacharjee A, Gupta AK. Development of a Low Cost Autonomous Car Parking System: Towards Smart City. In: and others, editor. Intelligent Electrical Systems: A Step towards Smarter Earth. 2021;p. 105–111. Available from: https://doi.org/10.1201/9780429355998-13.
3) Fahim A, Hasan M, Chowdhury MA. Smart parking systems: comprehensive review based on various aspects. *Heliyon*. 2021;7(5):1–21. Available from: https://doi.org/10.1016/j.heliyon.2021.e07050.
4) Karri S, Dhabu MM. Multistage Game Model Based Dynamic Pricing for Car Parking Slot to Control Congestion. *Sustainability*. 2022;14(19):1–15. Available from: https://doi.org/10.3390/su141911808.

5) Sarker VK, Gia TN, Dhaou IB, Westerlund T. Smart Parking System with Dynamic Pricing, Edge-Cloud Computing and LoRa. *Sensors*. 2020;20(17):1–22. Available from: https://doi.org/10.3390/s20174669.
6) Pham TN, Tsai MF, Nguyen DB, Dow CR, Deng DJ. A Cloud-Based Smart-Parking System Based on Internet-of-Things Technologies. *IEEE Access*. 2015;3:1581–1591. Available from: https://doi.org/10.1109/ACCESS.2015.2477299.
7) Poh LZ, Connie T, Ong TS, Goh MKO. Deep Reinforcement Learning-Based Dynamic Pricing for Parking Solutions. *Algorithms*. 2023;16(1):1–26. Available from: https://doi.org/10.3390/a16010032.
8) Bibi N, Majid MN, Dawood H, Guo P. Automatic Parking Space Detection System. In: and others, editor. 2nd International Conference on Multimedia and Image Processing (ICMIP). IEEE. 2017;p. 11–15. Available from: https://doi.org/10.1109/ICMIP.2017.4.
9) Kotb AO, Shen YC, Zhu X, Huang Y. iParker - A New Smart Car-Parking System Based on Dynamic Resource Allocation and Pricing. *IEEE Transactions on Intelligent Transportation Systems*. 2016;17(9):2637–2647. Available from: https://doi.org/10.1109/TITS.2016.2531636.
10) Mondal MA, Rehena Z, Janssen M. Smart parking management system with dynamic pricing. *Journal of Ambient Intelligence and Smart Environments*. 2021;13(6):473–494. Available from: https://doi.org/10.3233/AIS-210615.
11) Shangbin N, Zhenyu H, Yang Y, Zhenzhou Y, Xianyu W. CARSP: A Smart Parking System Based on Doubly Periodic Rolling Horizon Allocation Approach. *Journal of Advanced Transportation*. 2022;2022:1–18. Available from: https://doi.org/10.1155/2022/1373391.
12) ReactJS Documentation. . Available from: https://reactjs.org/docs/getting-started.html.
13) Semantic UI Documentation. . Available from: https://semantic-ui.com/introduction/build-tools.html.
14) MongoDB Documentation. . Available from: https://www.mongodb.com/docs/manual/tutorial/getting-started/.
15) Mapbox API Documentation. . Available from: https://docs.mapbox.com/api/overview/.
16) Mapbox API Documentation for Geocoding. . Available from: https://docs.mapbox.com/api/search/geocoding/.