



Μηχανική Λογισμικού Ι
Τομέας Ηλεκτρονικής και Υπολογιστών
Τμήμα ΗΜΜΥ
Α.Π.Θ

8^ο Εξάμηνο

Άνοιξη 2024



CurbSprings

Your Reliable Partner in Urban Mobility

Σχεδίαση και Ανάπτυξη Διεπαφών REST του Συστήματος

Del.1.3

Version 0.6

Πίττης Γεώργιος gkpittis@ece.auth.gr
Γουρδομιχάλης Αναστάσιος anasgour@ece.auth.gr
Τσαρναδέλης Αθανάσιος Γρηγόριος atsarnad@ece.auth.gr
Φωτιάδης Αλέξανδρος afotiadis@ece.auth.gr

30/05/2024



Ιστορικό Αλλαγών

Όνομα	Ημερομηνία	Αλλαγή	Έκδοση
A. Συμεωνίδης	17/05/2007	Δημιουργία εγγράφου. Προσαρμογή των προτύπων του K. E. Wiegers και του M. Smialek's.	0.1
A. Συμεωνίδης	29/3/2014	Μικρή αναθεώρηση – τροποποίηση ενοτήτων	0.1.3
X. Ζολώτας	10/4/2020	Μεγάλη αναθεώρηση – αφαίρεση ενοτήτων	0.4
X. Ζολώτας	15/4/2020	Μεγάλη αναθεώρηση – προσθήκη ενότητας REST προδιαγραφών	0.5.3
K. Παναγιώτου	25/4/2020	Μεγάλη αναθεώρηση – προσθήκη ενότητας Nodered περιγραφής	0.5.7
A. Συμεωνίδης	30/4/2020	Αναθεώρηση και τελική δομή προτύπου	0.6
K. Παναγιώτου	6/5/2024	Μεγάλη αναθεώρηση – διαγραφή ενότητας Nodered περιγραφής	0.7
K. Παναγιώτου	10/5/2024	Μικρή αναθεώρηση – Τροποποίηση ενοτήτων	0.7.1
A. Συμεωνίδης	15/5/2024	Αναθεώρηση και τελική δομή προτύπου	0.7.2

Μέλη της Ομάδας Ανάπτυξης

Όνομα	ΟΑ	Email
A. Συμεωνίδης	*	asymeon@issel.ee.auth.gr
Πίττης Γεώργιος	25	gkpittis@ece.auth.gr
Γουρδομιχάλης Αναστάσιος	25	anasgour@ece.auth.gr
Τσαρναδέλης Αθανάσιος Γρηγόριος	25	atsarnad@ece.auth.gr
Φωτιάδης Αλέξανδρος	25	afotiadis@ece.auth.gr



Πίνακας Περιεχομένων

Πίνακας Περιεχομένων	3
Λίστα Σχημάτων.....	4
1. Αναγνώριση Πόρων (Resources) Συστήματος.....	5
1.1. Πόροι One-Off.....	5
1.2. Πόροι Δεδομένων	5
1.3. Αλγοριθμικοί Πόροι	5
2. Τεκμηρίωση REST API.....	6
2.1. Πόρος Spot.....	6
2.1.1. Μοντέλο Δεδομένων	6
2.1.2. Τερματικό GET /spot.....	7
2.1.3. Τερματικό POST /spot.....	8
2.1.4. Τερματικό PUT /spot/{id}	9
2.1.5. Τερματικό DELETE /spot/{id}	10
2.1.6. Τερματικό GET /spot/search	10
2.2. Πόρος LicensePlate.....	12
2.2.1. Μοντέλο Δεδομένων	12
2.2.2. Τερματικό POST /licensePlate	12
2.2.3. Τερματικό PUT/licensePlate	13
2.3. Πόρος Reservation.....	15
2.3.1. Μοντέλο Δεδομένων	15
2.3.2. Τερματικό POST /reservation.....	16
2.3.3. Τερματικό PUT /reservation/{id}	17
2.3.4. Τερματικό DELETE /reservation/{id}	19
2.3.4.1 Παράμετροι εισόδου	19
2.4. Πόρος SpotOwner.....	20
2.4.1 Μοντέλο Δεδομένων	20
2.4.2 Τερματικό POST /spotowner.....	21
2.5. Πόρος Payment	21
2.5.1 Μοντέλο Δεδομένων	22
2.5.2 Τερματικό POST /payment.....	22



Λίστα Σχημάτων

Figure 1: Endpoints του πόρου ParkingSpot.....	6
Figure 2: Μοντέλο δεδομένων πόρου ParkingSpot.....	6
Figure 3 : Endpoints του πόρου LicensePlate	12
Figure 4 : Μοντέλο δεδομένων πόρου LicensePlate	12
Figure 5: Endpoints του πόρου Reservation.....	15
Figure 6: Μοντέλο δεδομένων του πόρου Reservation.....	15
Figure 7:Endpoint του πόρου SpotOwner	20
Figure 8: Μοντέλο δεδομένων του πόρου SpotOwner.....	20
Figure 9: Endpoint του πόρου Payment	21
Figure 10: Μοντέλο δεδομένων του πόρου Payment.....	22



1. Αναγνώριση Πόρων (Resources) Συστήματος

1.1. Πόροι One-Off

Κλάση BEC	Πόρος REST	Endpoints (HTTP Verbs)

1.2. Πόροι Δεδομένων

Κλάση BEC	Πόρος REST	Endpoints (HTTP Verbs)
ParkingSpot	/spot	GET
LicensePlate	/licensePlate	POST, PUT
Reservation	/reservation	POST
Reservation	/reservation/{id}	PUT, DELETE
ParkingSpot	/spot	POST
ParkingSpot	/spot/{id}	PUT, DELETE
SpotOwner	/spotowner	POST

1.3. Αλγοριθμικοί Πόροι

Κλάση BEC	Πόρος REST	Endpoints (HTTP Verbs)
ParkingSpot	/spot/search	GET
Payment	/payment	POST



2. Τεκμηρίωση REST API

2.1. Πόρος ParkingSpot

Spot Spot endpoints		^
GET	/spot Retrieve all available parking spots	▼
POST	/spot Add a new parking spot	▼
PUT	/spot/{id} Modify the information of a specific parking spot	▼
DELETE	/spot/{id} Remove a specific parking spot from the system	▼
GET	/spot/search Search for available parking spots	▼

Figure 1: Endpoints του πόρου ParkingSpot

Endpoint	User Story	FR
/spot	3, 4, 5, 7	1, 2, 12, 13, 14

Χρησιμοποιώ τον πόρο ParkingSpot για να διαχειριστώ ότι αφορά τις θέσεις πάρκινγκ του συστήματος μου. Μπορώ να δημιουργήσω, να τροποποιήσω, να διαγράψω, να πάρω όλες τις διαθέσιμες θέσεις καθώς και να ψάξω θέσεις με συγκεκριμένα κριτήρια.

2.1.1. Μοντέλο Δεδομένων

Ο πόρος ParkingSpot αφορά τις θέσεις πάρκινγκ και έχει ως χαρακτηριστικά τον μοναδικό αριθμό της θέσης σε μορφή integer, την διεύθυνση της θέσης σε μορφή string, το τύπο της θέσης επίσης σε μορφή string και την διαθεσιμότητα φορτιστή της θέσης σε μορφή Boolean.

```
ParkingSpot {  
  id integer  
  address string  
  type string  
  chargerAvailability boolean  
}
```

Figure 2: Μοντέλο δεδομένων πόρου ParkingSpot



2.1.2. Τερματικό GET /spot

Το endpoint αυτό επιστρέφει όλες τις διαθέσιμες θέσεις και ικανοποιεί το FR1, όπως φαίνεται παρακάτω.

2.1.2.1. Παράμετροι εισόδου

GET **/spot** Retrieve all available parking spots

FR1: The user must be able to view all available parking spots

Parameters

No parameters

2.1.2.2. Μοντέλο δεδομένων εξόδου

Το endpoint επιστρέφει μια λίστα αντικειμένων ParkingSpot, όπως φαίνεται στην εικόνα.

Responses

Code	Description	Links
200	A list of parking spots	No links

Media type

application/json

Controls Accept header.

Example Value | Schema

```
[
  {
    "id": 0,
    "address": "string",
    "type": "string",
    "chargerAvailability": true
  }
]
```



2.1.3. Τερματικό POST /spot

Το endpoint αυτό δημιουργεί μια νέα θέση ParkingSpot, και ικανοποιεί το FR12 όπως φαίνεται παρακάτω. Στο request body δίνονται τα απαραίτητα στοιχεία για να δημιουργηθεί ένα αντικείμενο ParkingSpot.

2.1.3.1 Παράμετροι εισόδου

POST /spot Add a new parking spot

FR12: The spot owner must be able to add a new parking spot to the system

Parameters

No parameters

Request body required

Example Value | Schema

```
{
  "id": 0,
  "address": "string",
  "type": "string",
  "chargerAvailability": true
}
```

2.1.3.2 Μοντέλο δεδομένων εξόδου

Αν τα στοιχεία της θέσης δοθούν σωστά όπως φαίνεται παραπάνω, το endpoint επιστρέφει το 201 Created.

Responses	
Code	Description
201	Parking spot added successfully



2.1.4. Τερματικό PUT /spot/{id}

Το endpoint παίρνει σαν παραμέτρους τα στοιχεία του ParkingSpot και τα τροποποιεί. Ικανοποιεί το FR13 όπως φαίνεται παρακάτω. Στο request body δίνονται τα απαραίτητα στοιχεία για να τροποποιηθεί ένα αντικείμενο ParkingSpot.

2.1.4.1 Παράμετροι εισόδου

PUT	/spot/{id}	Modify the information of a specific parking spot
FR13: The spot owner must be able to modify the information of their spots		
Parameters		
Name	Description	
id * required integer (path)	id	
address * required string (query)	address	
type * required string (query)	type	
charger * required boolean (query)	--	
Request body required		
Parking spot object to update		
Example Value Schema		
<pre>{ "id": 0, "address": "string", "type": "string", "chargerAvailability": true }</pre>		

2.1.4.2 Μοντέλο δεδομένων εξόδου

Το endpoint επιστρέφει τον κωδικό 200 αν η τροποποίηση της θέσης πάρκινγκ ήταν επιτυχής.

Responses	
Code	Description
200	Parking spot updated successfully



2.1.5. Τερματικό DELETE /spot/{id}

Το endpoint παίρνει σαν παράμετρο τον μοναδικό αριθμό(id) μιας θέσης πάρκινγκ και την διαγράφει από το σύστημα. Ικανοποιεί την FR14 όπως φαίνεται παρακάτω.

2.1.5.1 Παράμετροι εισόδου

DELETE	/spot/{id}	Remove a specific parking spot from the system
FR14: The spot owner must be able to remove a spot from the system		
Parameters		
Name	Description	
id * required	<input type="text" value="id"/>	
integer		
(path)		

2.1.5.2 Μοντέλο δεδομένων εξόδου

Το endpoint επιστρέφει κωδικό 204 αν η διαγραφή της θέσης πάρκινγκ είναι επιτυχής.

Responses		
Code	Description	Links
204	Parking spot removed successfully	No links

2.1.6. Τερματικό GET /spot/search

Το endpoint παίρνει σαν παραμέτρους τα κριτήρια αναζήτησης θέσης πάρκινγκ και βρίσκει διαθέσιμες θέσεις πάρκινγκ που πληρούν αυτά τα κριτήρια. Ικανοποιεί το FR2 όπως φαίνεται παρακάτω.



2.1.6.1 Παράμετροι εισόδου

GET	/spot/search	Search for available parking spots
FR2: The user must be able to search for available parking spots in the system		
Parameters		
Name	Description	
address * required string (query)	<input type="text" value="address"/>	
type * required string (query)	<input type="text" value="type"/>	
charger * required boolean (query)	<input type="checkbox" value=""/>	

2.1.6.2 Μοντέλο δεδομένων εξόδου

Το endpoint επιστρέφει μια λίστα αντικειμένων ParkingSpot που πληρούν τα κριτήρια αναζήτησης.

Code	Description	Links
200	A list of available parking spots matching the search criteria	No links
Media type <input type="text" value="application/json"/>		
Controls Accept header.		
Example Value Schema		
<pre>[{ "id": 0, "address": "string", "type": "string", "chargerAvailability": true }]</pre>		



2.2. Πόρος LicensePlate

LicensePlate LicensePlate endpoints

POST	/licensePlate	Register vehicle's license plate	▼
PUT	/licensePlate	Modify vehicle's license plate	▼

Figure 3 : Endpoints του πόρου LicensePlate

Endpoint	User Story	FR
/licensePlate	9,10	4,5

Χρησιμοποιώ τον πόρο LicensePlate για να διαχειριστώ την πινακίδα του οχήματος του χρήστη. Ο χρήστης μπορεί να καταχωρήσει την πινακίδα του μέσα στο σύστημα. Επίσης, μπορεί να επεξεργαστεί την ήδη καταχωρημένη πινακίδα του.

2.2.1. Μοντέλο Δεδομένων

Ο πόρος LicensePlate αφορά την πινακίδα του χρήστη. Έχει ως χαρακτηριστικά τον μοναδικό αριθμό του χρήστη που κατέχει την πινακίδα σε μορφή integer καθώς και το όνομα της πινακίδας σε μορφή string.

```
LicensePlate {  
  userId          integer  
  licensePlate    string  
}
```

Figure 4 : Μοντέλο δεδομένων πόρου LicensePlate

2.2.2. Τερματικό POST /licensePlate

Το endpoint δημιουργεί την πινακίδα του χρήστη μέσα στο σύστημα και ικανοποιεί το FR4 όπως φαίνεται παρακάτω. Στο request body δίνονται τα απαραίτητα στοιχεία για να δημιουργηθεί ένα αντικείμενο LicensePlate.



2.2.2.1 Παράμετροι εισόδου

POST `/licensePlate` Register vehicle's license plate

FR4: The user must be able to register their vehicle's license plate in the system

Parameters

No parameters

Request body required

License plate to register

Example Value | **Schema**

```
{
  "userId": 0,
  "licensePlate": "string"
}
```

2.2.2.2 Μοντέλο δεδομένων εξόδου

Το endpoint επιστρέφει κωδικό 201 αν η καταχώρηση της νέας πινακίδας μέσα στο σύστημα είναι επιτυχής.

Responses		
Code	Description	Links
201	License plate registered successfully	No links

2.2.3. Τερματικό PUT/licensePlate

Το endpoint τροποποιεί την ήδη καταχωρημένη πινακίδα του χρήστη μέσα στο σύστημα και ικανοποιεί το FR5 όπως φαίνεται παρακάτω. Στο request body δίνονται τα απαραίτητα στοιχεία για να τροποποιηθεί το αντικείμενο LicensePlate.



2.2.3.1 Παράμετροι εισόδου

PUT `/licensePlate` Modify vehicle's license plate

FR5: The user must be able to modify their vehicle's license plate in the system

Parameters

No parameters

Request body *required*

License plate object to update

Example Value | Schema

```
{
  "userId": 0,
  "licensePlate": "string"
}
```

2.2.3.2 Μοντέλο δεδομένων εξόδου

Το endpoint επιστρέφει κωδικό 200 αν η πινακίδα τροποποιήθηκε με επιτυχία.

Responses		
Code	Description	Links
200	License plate updated successfully	No links



2.3. Πόρος Reservation

Reservation <small>Reservation endpoints</small>			^
POST	/reservation	Create a new reservation	▼
PUT	/reservation/{id}	Modify a specific reservation	▼
DELETE	/reservation/{id}	Cancel a specific reservation	▼

Figure 5: Endpoints του πόρου Reservation

Endpoint	User Story	FR
/reservation	1,2	6,7

Χρησιμοποιώ τον πόρο Reservation για να διαχειριστώ τις κρατήσεις που κάνει ο χρήστης του συστήματός μου. Ο χρήστης μπορεί να δημιουργήσει μια νέα κράτηση αλλά και να τροποποιήσει ή/και ακυρώσει κρατήσεις που έχει ήδη κάνει.

2.3.1. Μοντέλο Δεδομένων

Ο πόρος Reservation αφορά τις κρατήσεις που κάνει ο χρήστης και έχει ως χαρακτηριστικά τον μοναδικό αριθμό με τον οποίο καταχωρείται στο σύστημα ως integer, τον μοναδικό αριθμό της θέσης σε μορφή integer, το id του χρήστη που έχει κάνει την κράτηση σε μορφή integer, την ώρα έναρξης και λήξης της κράτησης σε μορφή string και DataType date-time και τέλος την ημερομηνία της κράτησης σε μορφή string και DataType date.

```
Reservation {  
  id                integer  
  spotId            integer  
  userId            integer  
  startTime         string($date-time)  
  duration          string($date-time)  
  date              string($date)  
}
```

Figure 6: Μοντέλο δεδομένων του πόρου Reservation



2.3.2. Τερματικό POST /reservation

Το endpoint δημιουργεί την κράτηση του χρήστη στο σύστημα και ικανοποιεί το FR6 όπως φαίνεται παρακάτω. Στο request body δίνονται τα απαραίτητα στοιχεία για να δημιουργηθεί ένα αντικείμενο Reservation.

2.3.2.1 Παράμετροι εισόδου

POST **/reservation** Create a new reservation ^

FR6: The user must be able to reserve a parking spot

Parameters Try it out

No parameters

Request body required application/json

Reservation object to create

Example Value | Schema

```
{
  "id": 0,
  "spotId": 0,
  "userId": 0,
  "startTime": "2024-05-25T15:25:31.172Z",
  "duration": "2024-05-25T15:25:31.172Z",
  "date": "2024-05-25"
}
```

2.3.2.2 Μοντέλο δεδομένων εξόδου

Responses		
Code	Description	Links
201	Reservation created successfully	No links

Το endpoint επιστρέφει κωδικό 201 αν η κράτηση έγινε με επιτυχία.



2.3.3. Τερματικό PUT /reservation/{id}

Το endpoint τροποποιεί την κράτηση του χρήστη στο σύστημα και ικανοποιεί το FR7 όπως φαίνεται παρακάτω. Το endpoint παίρνει σαν παραμέτρους εισόδου τα στοιχεία ενός ήδη υπάρχοντος αντικειμένου Reservation. Στο request body δίνονται τα απαραίτητα στοιχεία για να τροποποιηθεί ένα ήδη υπάρχον αντικείμενο Reservation.

2.3.3.1 Παράμετροι εισόδου

PUT	/reservation/{id}	Modify a specific reservation
FR7: The user must be able to manage their reservation in the system		
Parameters		
Name	Description	
id * required integer (path)	id	
spotId * required integer (query)	spotId	
userId * required integer (query)	userId	
startTime * required string(\$date-time) (query)	startTime	
duration * required string(\$date-time) (query)	duration	
date * required string(\$date) (query)	date	



Request body required

Reservation object to update

Example Value | Schema

```
{
  "id": 0,
  "spotId": 0,
  "userId": 0,
  "startTime": "2024-05-25T15:29:28.766Z",
  "duration": "2024-05-25T15:29:28.766Z",
  "date": "2024-05-25"
}
```

2.3.3.2 Μοντέλο δεδομένων εξόδου

Responses	
Code	Description
200	Reservation updated successfully

Το endpoint επιστρέφει κωδικό 200 αν η κράτηση τροποποιήθηκε επιτυχώς.



2.3.4. Τερματικό DELETE /reservation/{id}

Το endpoint διαγράφει την κράτηση του χρήστη στο σύστημα και ικανοποιεί το FR7 όπως φαίνεται παρακάτω. Το endpoint παίρνει σαν παράμετρο εισόδου το id της κράτησης (δηλαδή το id του αντικειμένου Reservation) που θέλω να διαγράψω.

2.3.4.1 Παράμετροι εισόδου

DELETE /reservation/{id} Cancel a specific reservation ^

FR7: The user must be able to manage their reservation in the system

Parameters [Try it out](#)

Name	Description
id * required	
integer	id
(path)	

2.3.4.2 Μοντέλο δεδομένων εξόδου

Responses		
Code	Description	Links
204	Reservation cancelled successfully	No links

Το endpoint επιστρέφει κωδικό 204 αν η κράτηση διαγράφηκε επιτυχώς.



2.4. Πόρος SpotOwner

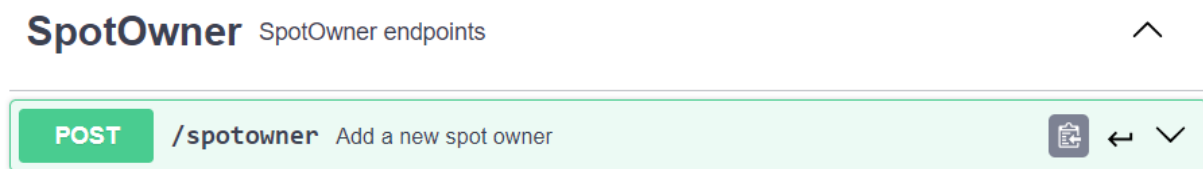


Figure 7: Endpoint του πόρου SpotOwner

Endpoint	User Story	FR
/spotowner	6	15

Χρησιμοποιώ τον πόρο SpotOwner για να διαχειριστώ τους ιδιοκτήτες των θέσεων πάρκινγκ του συστήματός μου. Μπορώ να προσθέτω έναν νέο ιδιοκτήτη θέσης πάρκινγκ.

2.4.1 Μοντέλο Δεδομένων

Ο πόρος SpotOwner έχει ως χαρακτηριστικά τον μοναδικό αριθμό του κάθε ιδιοκτήτη σε μορφή integer, το όνομα του σε μορφή string, το email του σε μορφή string, τον αριθμό ταυτότητας του σε μορφή string, τον αριθμό τηλεφώνου του σε μορφή αριθμού (number), και μια λίστα από θέσεις parking που του ανήκουν.



Figure 8: Μοντέλο δεδομένων του πόρου SpotOwner



2.4.2 Τερματικό POST /spotowner

Το endpoint αυτό δημιουργεί έναν νέο spotowner και ικανοποιεί το FR15 όπως φαίνεται παρακάτω.

2.4.2.1 Παράμετροι εισόδου

POST /spotowner Add a new spot owner ^

FR15: The system administrator must be able to add a spot owner to the system

Parameters [Try it out](#)

No parameters

2.4.2.2 Μοντέλο δεδομένων εξόδου

Responses		
Code	Description	Links
201	Spot owner added successfully	No links

Το endpoint επιστρέφει κωδικό 201 αν η δημιουργία του νέου spotowner μέσα στο σύστημα ήταν επιτυχής.

2.5. Πόρος Payment

Payment Payment endpoints ^

POST /payment Process a payment through the payment gateway v

Figure 9: Endpoint του πόρου Payment



Endpoint	User Story	FR
/payment	8	8

Χρησιμοποιούμε τον πόρο Payment για να διαχειριστούμε την πληρωμή του χρήστη.

2.5.1 Μοντέλο Δεδομένων

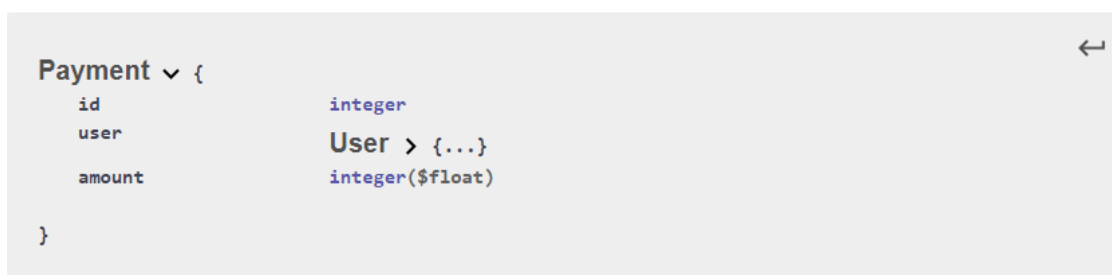


Figure 10: Μοντέλο δεδομένων του πόρου Payment

Ο πόρος Payment έχει ως χαρακτηριστικά τον χρήστη(αντικείμενο User) που εκτελεί την πληρωμή , τον μοναδικό αριθμό του χρήστη σε μορφή integer, και το ποσό της πληρωμής(επιτρέπει και integer και float τιμή).

2.5.2 Τερματικό POST /payment

Το endpoint αυτό δημιουργεί μια νέα πληρωμή (ένα νέο αντικείμενο Payment) και ικανοποιεί το FR8 όπως φαίνεται παρακάτω. Στο request body δίνονται τα απαραίτητα στοιχεία για να δημιουργηθεί ένα νέο αντικείμενο Payment.

2.5.2.1 Παράμετροι εισόδου

POST /payment Process a payment through the payment gateway

FR8: The user must be able to make a payment through the payment gateway

Parameters

Try it out

No parameters



Request body required

application/json

Payment object to process

Example Value | Schema

```
{
  "id": 0,
  "user": {
    "id": 0,
    "name": "string",
    "licensePlate": {
      "userId": 0,
      "licensePlate": "string"
    }
  },
  "reservation": [
    {
      "id": 0,
      "spotId": 0,
      "userId": 0,
      "startTime": "2024-05-30T15:58:03.396Z",
      "duration": "2024-05-30T15:58:03.396Z",
      "date": "2024-05-30"
    }
  ]
},
  "amount": 0
}
```

2.5.2.2 Μοντέλο δεδομένων εξόδου

Responses		
Code	Description	Links
201	Payment processed successfully	No links

Επιτυχής πληρωμή με επιστρεφόμενη τιμή 201 από το endpoint.