## AutoPartsService API Documentation

### **LoginController**

#### **1.1 POST** **/login**

**Description**: Endpoint for sending a login SMS to the user.

**Payload**: A JSON containing the phone number of the user.

**Example**:

json

{

"phoneNumber": "+12345678901"

}

**Response**: A JSON containing the status of the request, a message, and the details of the login SMS.

**Example**:

json

{

"success": true,

"message": "SMS sent.",

"data": {

"phoneNumber": "+12345678901",

"creationDate": "2023-06-07T18:47:30.711Z",

"newUser": false,

"smsCode": "123456"

}

}

**Status Codes**:

* 200 OK if the SMS is sent successfully.
* 400 Bad Request if the phone number is not valid.
* 500 Internal Server Error if there was a problem with sending the SMS.

#### **1.2 POST** **/login/authenticate**

**Description**: Endpoint for authenticating a user using the SMS code.

**Payload**: A JSON containing the phone number of the user and the SMS code.

**Example**:

json

{

"phoneNumber": "+12345678901",

"smsCode": "123456"

}

**Response**: A JSON containing the status of the request, a message, and the JWT token if the authentication is successful.

**Example**:

json

{

"success": true,

"message": "User found.",

"data": {

"Token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",

"User": {

"id": 1,

"name": "John Doe",

"email": "john.doe@example.com",

"phoneNumber": "+12345678901",

"registrationDate": "2023-06-07T18:47:30.711Z",

"password": "\*\*\*\*\*\*\*\*",

"address": "123 Main St, Anytown, Anystate"

}

}

}

**Status Codes**:

* 200 OK if the authentication is successful.
* 400 Bad Request if the phone number or the SMS code is not valid.
* 404 Not Found if the user does not exist.

#### **1.3 POST** **/login/register**

**Description**: Endpoint for registering a new user.

**Payload**: A JSON containing the user's details.

**Example**:

json

{

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",

"phoneNumber": "+12345678901",

"userType": "Common",

"name": "John Doe",

"email": "john.doe@example.com",

"password": "Password1234",

"country": "USA",

"region": "NY",

"city": "New York City",

"street": "123 Main St"

}

**Response**: A JSON containing the status of the request, a message, and the user's details if the registration is successful.

**Example**:

json

{

"success": true,

"message": "User registered.",

"data": {

"id": 1,

"name": "John Doe",

"email": "john.doe@example.com",

"phoneNumber": "+12345678901",

"password": "\*\*\*\*\*\*\*\*",

"registrationDate": "2023-06-07T19:03:24.981Z",

"address": {

"country": "USA",

"region": "NY",

"city": "New York City",

"street": "123 Main St"

}

}

}

**Status Codes**:

* 200 OK if the registration is successful.
* 400 Bad Request if the user's details are not valid.
* 409 Conflict if the user already exists.
* 500 Internal Server Error if there was a problem with the registration.

#### **1.4 GET** **/login/userInfo**

**Description**: Endpoint for retrieving user information.

**Query parameters**: jwt (the user's JWT token) and deviceId (the user's device ID).

**Response**: A JSON containing the status of the request, a message, and the user's details if the user is found.

**Example**:

json

{

"success": true,

"message": "User found.",

"data": {

"id": 1,

"name": "John Doe",

"email": "john.doe@example.com",

"phoneNumber": "+12345678901",

"password": "\*\*\*\*\*\*\*\*",

"registrationDate": "2023-06-07T19:03:24.981Z",

"address": {

"country": "USA",

"region": "NY",

"city": "New York City",

"street": "123 Main St"

}

}

}

**Status Codes**:

* 200 OK if the user is found.
* 400 Bad Request if the JWT token or the device ID is not valid.
* 404 Not Found if the user does not exist.

## 2. RequestController

#### **2.1 POST** **/api/request/{userCommonId}**

**Description**: Endpoint for creating a new request.

**URL Parameter**: userCommonId (the ID of the common user)

**Payload**: A JSON containing the details of the request to be created.

**Example**:

json

{

"Header": "Need parts for a 2009 Honda Civic",

"Description": "Looking for a new transmission and radiator",

"Price": 500

}

**Response**: A JSON containing the status of the request and a message.

**Example**:

json

{

"Success": true,

"Message": "Request created successfully."

}

**Status Codes**:

* 200 OK if the request is created successfully.
* 400 Bad Request if the request details are not valid.
* 404 Not Found if the user does not exist.

#### **2.2 GET** /**api/request/active**

**Description**: Endpoint for retrieving all active requests.

**Response**: A JSON containing the status of the request, a message, and a list of all active requests.

**Example**:

json

{

"Success": true,

"Message": "Active requests retrieved successfully.",

"Data": [

{

"RequestId": 1,

"Header": "Need parts for a 2009 Honda Civic",

"Description": "Looking for a new transmission and radiator",

"Price": 500,

"UserCommonId": 1,

"CreationDate": "2023-06-07T18:47:30.711Z",

"Active": true

},

// More requests...

]

}

**Status Codes**:

* 200 OK if the active requests are retrieved successfully.

#### **2.3 POST** **/api/request/accept/{requestId}/{userId}**

**Description**: Endpoint for a user to accept a request.

**URL Parameters**: requestId (the ID of the request) and userId (the ID of the user)

**Response**: A JSON containing the status of the request and a message.

**Example**:

json

{

"Success": true,

"Message": "Request accepted successfully."

}

**Status Codes**:

* 200 OK if the request is accepted successfully.
* 400 Bad Request if the request or user ID is not valid.
* 404 Not Found if the request or the user does not exist.

## UserController

### **3.1 POST /user/addCar**

Endpoint for adding a new car.

URL Parameter: userCommonId (integer - ID of the user to add the car)

Payload: A JSON containing the details of the car to be added.

Example:

json

{

"Make": "Honda",

"Model": "Civic",

"Color": "Red",

"StateNumber": "ABC 123",

"VinNumber": "1HGCM82633A123456",

"BodyNumber": "123456789"

}

Response: A JSON containing the status of the operation, a message, and the added car.

Example:

json

{

"Success": true,

"Message": "Car added successfully.",

"Data": {

"Make": "Honda",

"Model": "Civic",

"Color": "Red",

"StateNumber": "ABC 123",

"VinNumber": "1HGCM82633A123456",

"BodyNumber": "123456789",

"UserCommonId": 1

}

}

Status Codes:

200 OK if the car is added successfully.

400 Bad Request if the car details are not valid.

404 Not Found if the user does not exist.

### **3.2 PUT /user/UserCommon/{id}**

Endpoint for updating a user.

URL Parameter: id (integer - ID of the user to be updated)

Payload: A JSON containing the details to be updated.

Example:

json

{

"Name": "John Doe",

"Email": "john.doe@example.com",

"PhoneNumber": "1234567890",

"Password": "password",

"Address": {

"Country": "Country",

"Region": "Region",

"City": "City",

"Street": "Street"

}

}

Response: A JSON containing the status of the operation, a message, and the updated user.

Example:

json

{

"Success": true,

"Message": "UserCommon edited successfully.",

"Data": {

"Id": 1,

"Name": "John Doe",

"Email": "john.doe@example.com",

"PhoneNumber": "1234567890",

"RegistrationDate": "2023-06-07T18:47:30.711Z",

"Password": "password",

"Address": {

"Country": "Country",

"Region": "Region",

"City": "City",

"Street": "Street"

},

"Cars": [

// List of cars...

]

}

}

Status Codes:

200 OK if the user is updated successfully.

400 Bad Request if the user details are not valid.

404 Not Found if the user does not exist.

### **3.3 DELETE /user/delete/{id}**

Endpoint for deleting a user.

URL Parameter: id (integer - ID of the user to be deleted)

Response: A JSON containing the status of the operation and a message.

Example:

json

{

"Success": true,

"Message": "User deleted successfully."

}

Status Codes:

200 OK if the user is deleted successfully.

400 Bad Request if the user ID is not valid.

404 Not Found if the user does not exist.

### **3.4 GET /user/details/{id}**

Endpoint for retrieving the details of a user.

URL Parameter: id (integer - ID of the user)

Response: A JSON containing the status of the operation, a message, and the user details.

Example:

json

{

"Success": true,

"Message": "User details retrieved successfully.",

"Data": {

"Id": 1,

"Name": "John Doe",

"Email": "john.doe@example.com",

"PhoneNumber": "1234567890",

"RegistrationDate": "2023-06-07T18:47:30.711Z",

"Address": {

"Country": "Country",

"Region": "Region",

"City": "City",

"Street": "Street"

},

"Cars": [

// List of cars...

]

}

}

Status Codes:

200 OK if the user details are retrieved successfully.

404 Not Found if the user does not exist.

### **3.5 GET /user/all**

Endpoint for retrieving all users.

Response: A JSON containing the status of the operation, a message, and a list of all users.

Example:

json

{

"Success": true,

"Message": "Users retrieved successfully.",

"Data": [

{

"Id": 1,

"Name": "John Doe",

"Email": "john.doe@example.com",

"PhoneNumber": "1234567890",

"RegistrationDate": "2023-06-07T18:47:30.711Z",

"Address": {

"Country": "Country",

"Region": "Region",

"City": "City",

"Street": "Street"

},

"Cars": [

// List of cars...

]

},

// More users...

]

}

Status Codes:

200 OK if the users are retrieved successfully.