SDE Intern - Backend

Assignment Solution Documentation

Introduction:

This is going to be the documentation for the assignment. Here I will provide a little bit of an introduction regarding the assignment and a small overview of my approach. The solution is implemented in TypeScript with NestJS as the framework. I have used maps for efficient addition of a car in the parking lot so that the nearest slot is always assigned to the car. I made it such that only O(n) space will be utilized for storing the car details and few constant space O(1) variables to keep track of the total slots count. The code is complemented with unit tests as well for all the services. I have deployed the code at two places firstly on Vercel and second on my own server just to reduce the chances of any last minute glitch. Now let's move to the API details.

API Endpoints:

Github - https://github.com/ujjwalravi/finmo_backend
BASE_URL - https://finmo-backend-y5zi.vercel.app
Alternate BASE_URL - http://194.195.116.249:3000
Postman -
https://github.com/ujjwalravi/finmo_backend/tree/main/postman_collection

1. **GET /** - This is the homepage. You can visit this for a small overview about the endpoints. So, no need to visit the documentation again and again .
2. **POST /parking_lot -** *Initialize slots* - This is used to initialize the slots. This will set up a new parking slot. This can be used only once as initialization is a single time affair. To increase the slots a different method can be used at the same endpoint.

body	**response**
{ "no_of_slot": 3	{ "total_slot": 3

}	}

Different errors with different messages will be sent as response depending upon different requests like if tried to re-initialize or negative number as “no_of_slot”.

3. **PATCH /parking_lot -** *Increment slots* - This is used to increment the slots. Make sure that the slots are already initialized or else you will receive error messages as response.

body	**response**

{
"increment_slot": 2
}
	{
	"total_slot": 5
	}

Different errors with different messages will be sent as response depending upon different requests like if tried to re-initialize or negative number as “no_of_slot”.

4. **POST /park -** *Park the car in the nearest slot* - This is used to park the car in the nearest available slot. Passing both parameters is mandatory.

body	**response**

{
"car_reg_no": "KA-1234",
"car_color": "yellow"
}
	{
	"allocated_slot_number": 3
	}

Different errors will be sent depending upon requests like not passing both the parameters.

5. **GET /registration_numbers/:color -** *Fetch reg numbers based on color* - This is used to filter out and get the list of registration numbers based on the color given in the query parameter.

   If the provided query value doesn’t match with any car parked in the slots then it will return an empty list [ ].

query	response
/registration_numbers/yellow	["BR-312", "KA-312", "JH-312"]

6. **GET /slot_numbers/:color -** *Fetch slot numbers based on color* - This is used to filter out and get the list of slot numbers based on the color given in the query parameter.

   If the provided query value doesn't match with any car parked in the slots then it will return an empty list [ ].

query	response
/slot_numbers/yellow	[3, 2, 7]

7. **POST /park/clear -** *Free a parking slot* - This is used to free the slot occupied by a car. There are two ways one can free the slot i.e. either you pass the car registration number or the slot number.

body	response

{
"car_registration_no": "BR-312"
}
	{
	"freed_slot_number": 3
	}

body	response

{
"slot_number": 1
}
	{
	"freed_slot_number": 1
	}

   Appropriate error message will be sent as a response depending upon

requests.

8. **GET /park/status -** *Fetches the parked cars in the parking lot* - This will fetch all the occupied slots in the parking slot. If no car is parked then the empty list will be returned as a response.

response	<pre>[
	  {
	  "allocated_slot_number": 1,
	  "car_reg_no": "BR-312",
	  "color": "yellow"
	  },
	  {
	  "allocated_slot_number": 2,
	  "car_reg_no": "KA-312",
	  "color": "red"
	  }
	]</pre>

9. **GET /park/details -** *Fetches the slots details* - This will fetch the number of free, occupied and total slots.

response	<pre>{
	  "total_slots": 3,
	  "occupied_slots": 2,
	  "free_slots": 1
	}</pre>