

NAME	Uday Pal
UID	23BCS12013
CLASS	607-B

## ➤ NodeJs PRACTISE 2

- CODE

```

const express = require('express');
const app = express();
const port = 3000;

app.use(express.json());

let cards = [
  { "id": 1, "suit": "Hearts", "value": "Ace" },
  { "id": 2, "suit": "Spades", "value": "King" },
  { "id": 3, "suit": "Diamonds", "value": "Queen" }
];

let nextId = cards.length + 1;

app.get('/cards', (req, res) => {
  res.status(200).json(cards);
});

app.get('/cards/:id', (req, res) => {
  const id = parseInt(req.params.id);
  const card = cards.find(c => c.id === id);

  if (card) {
    res.status(200).json(card);
  }
});

```

```
    } else {
      res.status(404).json({ message: `Card with ID ${id} not found` });
    }
  });

app.post('/cards', (req, res) => {
  const { suit, value } = req.body;

  if (!suit || !value) {
    return res.status(400).json({ message: "Missing required fields: suit and value" });
  }

  const newCard = {
    id: nextId++,
    suit,
    value
  };

  cards.push(newCard);

  res.status(201).json(newCard);
});

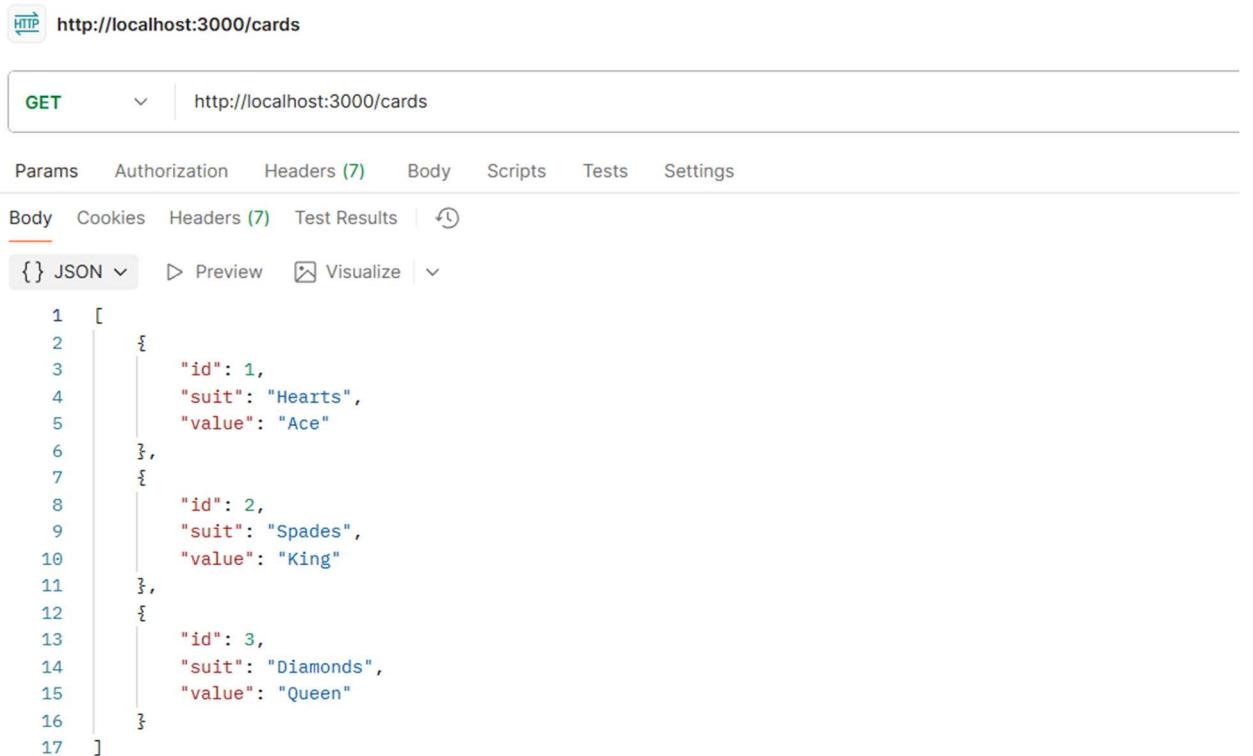
app.delete('/cards/:id', (req, res) => {
  const id = parseInt(req.params.id);
  const cardIndex = cards.findIndex(c => c.id === id);

  if (cardIndex !== -1) {
    const [deletedCard] = cards.splice(cardIndex, 1);

    res.status(200).json({
      message: `Card with ID ${id} removed`,
      card: deletedCard
    });
  } else {
    res.status(404).json({ message: `Card with ID ${id} not found` });
  }
});

app.listen(port, () => {
  console.log(`Card API listening at http://localhost:${port}`);
});
```

## ● OUTPUT



HTTP <http://localhost:3000/cards>

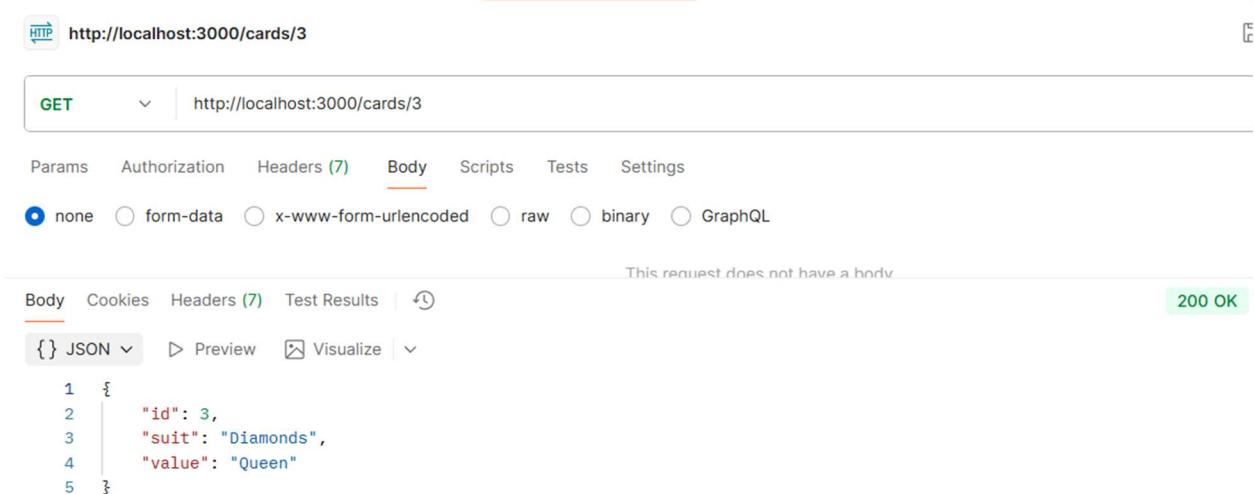
GET <http://localhost:3000/cards>

Params Authorization Headers (7) Body Scripts Tests Settings

Body Cookies Headers (7) Test Results

{ } JSON ▾ ▶ Preview Visualize ▾

```
1 [  
2   {  
3     "id": 1,  
4     "suit": "Hearts",  
5     "value": "Ace"  
6   },  
7   {  
8     "id": 2,  
9     "suit": "Spades",  
10    "value": "King"  
11  },  
12  {  
13    "id": 3,  
14    "suit": "Diamonds",  
15    "value": "Queen"  
16  }  
17 ]
```



HTTP <http://localhost:3000/cards/3>

GET <http://localhost:3000/cards/3>

Params Authorization Headers (7) Body Scripts Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL

This request does not have a body.

200 OK

Body Cookies Headers (7) Test Results

{ } JSON ▾ ▶ Preview Visualize ▾

```
1 {  
2   "id": 3,  
3   "suit": "Diamonds",  
4   "value": "Queen"  
5 }
```

HTTP <http://localhost:3000/cards>

POST <http://localhost:3000/cards>

Params Authorization Headers (9) **Body** Scripts Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL [JSON](#) ▾

```
1 {  
2   "id": 4,  
3   "suit": "Club",  
4   "value": "Jack"  
5 }
```

Body Cookies Headers (7) Test Results | ⏪ 201 Create

{ } JSON ▾ ▶ Preview Visualize ▾

```
1 {  
2   "id": 4,  
3   "suit": "Club",  
4   "value": "Jack"  
5 }
```

HTTP <http://localhost:3000/cards/2>

DELETE <http://localhost:3000/cards/2>

Params Authorization Headers (7) **Body** Scripts Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL

This request does not have a body.

Body Cookies Headers (7) Test Results | ⏪ 200 OK

{ } JSON ▾ ▶ Preview Visualize ▾

```
1 {  
2   "message": "Card with ID 2 removed",  
3   "card": {  
4     "id": 2,  
5     "suit": "Spades",  
6     "value": "King"  
7   }  
8 }
```

Output Link :- <http://localhost:3000/cards>

API's: [GET http://localhost:3000/cards](#) [GET http://localhost:3000/cards/3](#)

[POST http://localhost:3000/cards](#) [DELETE http://localhost:3000/cards/2](#)

