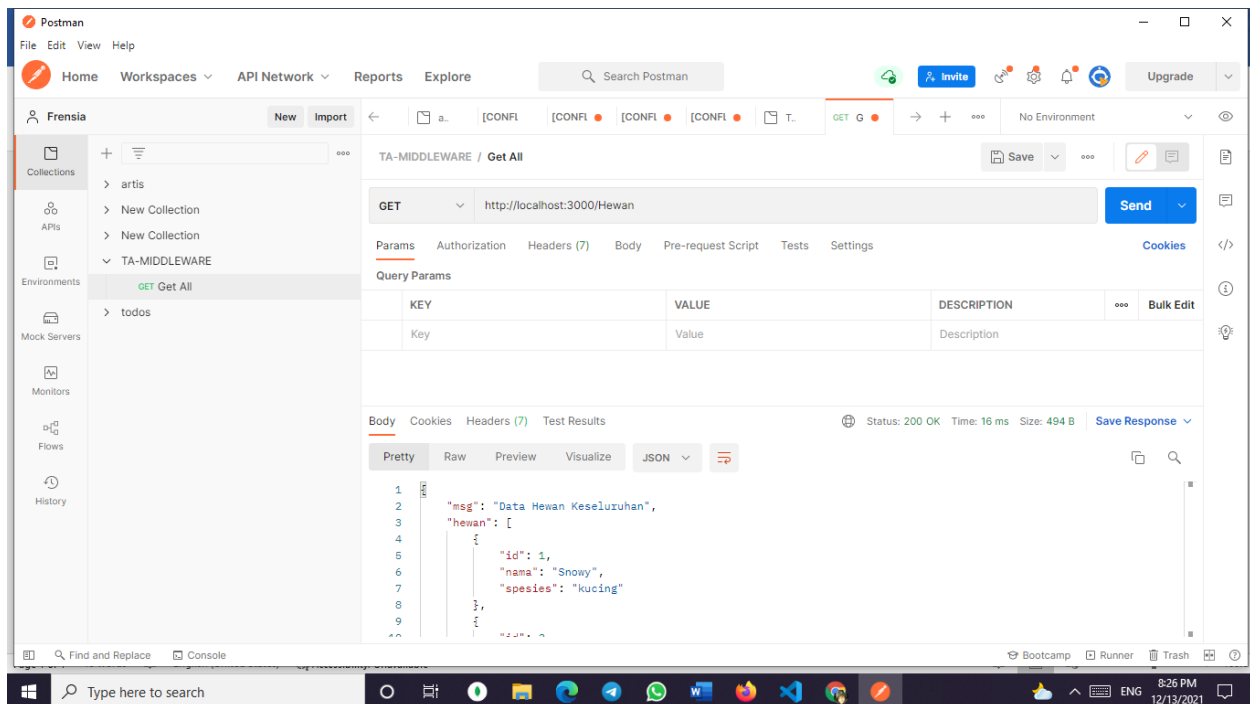


Nama : Frensia Tanaga Anaclaudia

Grup : 15

Tugas: Express Routing and Middleware

-Get All Hewan



-Get by ID

The screenshot shows the Postman application interface. On the left sidebar, the 'Collections' panel is open, showing a collection named 'TA-MIDDLEWARE' with a sub-collection 'Get by ID'. The main panel displays a GET request to the URL 'http://localhost:3000/Hewan/0/Snowy'. The 'Params' tab is selected, showing a table with columns 'KEY', 'VALUE', and 'DESCRIPTION'. The 'Body' tab is also visible, showing a JSON response in 'Pretty' format. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 18 ms', and 'Size: 313 B'.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 {
2   "msg": "Berikut Data Hewan",
3   "data": {
4     "id": 1,
5     "nama": "Snowy",
6     "spesies": "kucing"
7   }
8 }
```

-Put Hewan (Update data)

The screenshot shows the Postman application interface. On the left sidebar, the 'Collections' panel is open, showing a collection named 'TA-MIDDLEWARE' with a sub-collection 'Update Hewan'. The main panel displays a PUT request to the URL 'http://localhost:3000/Hewan/4'. The 'Body' tab is selected, showing a JSON request in 'Raw' format. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 17 ms', and 'Size: 484 B'.

```
1 {
2   "namaHewan": "cangmorulis",
3   "spesies": "biawak"
4 }
```

```
16 {
17   "spesies": "kucing"
18 },
19 {
20   "id": 4,
21   "nama": "cangmorulis",
22   "spesies": "biawak"
23 },
24 {
25   "id": 5,
26   "nama": "cangmorulis",
27   "spesies": "biawak"
28 }
```

-Post

The screenshot shows the Postman application interface. On the left sidebar, the 'Collections' panel is open, showing a collection named 'TA-MIDDLEWARE' with a sub-collection 'POST POST Hewan' selected. The main workspace displays a POST request to 'http://localhost:3000/Hewan'. The 'Body' tab is active, showing a JSON payload:

```
{  "namaHewan": "Hiu",  "namaSpesies": "Omnivora"}
```

. The 'Send' button is visible. Below the request, the 'Test Results' section shows a status of 200 OK, with a time of 67 ms and a size of 544 B. The response body is displayed in the 'Pretty' view, showing a JSON object:

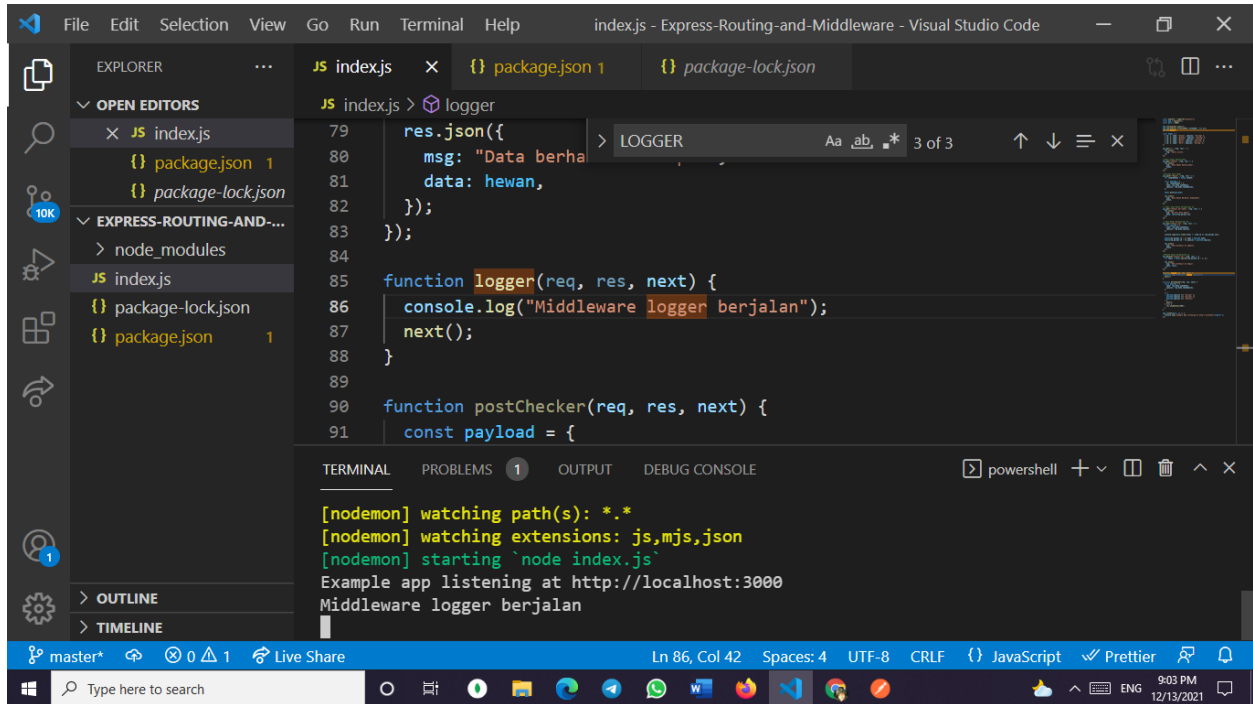
```
{  "id": 6,  "nama": "Hiu",  "spesies": "Omnivora"}
```

-Delete Hewan by ID

The screenshot shows the Postman application interface. On the left sidebar, the 'Collections' panel is open, showing a collection named 'TA-MIDDLEWARE' with a sub-collection 'DELETE Delete Hewan' selected. The main workspace displays a DELETE request to 'http://localhost:3000/Hewan/3'. The 'Send' button is visible. Below the request, the 'Test Results' section shows a status of 200 OK, with a time of 19 ms and a size of 493 B. The response body is displayed in the 'Pretty' view, showing a JSON object:

```
{  "id": 2,  "nama": "Blacki",  "spesies": "anjing"}
```

-Middleware Logger

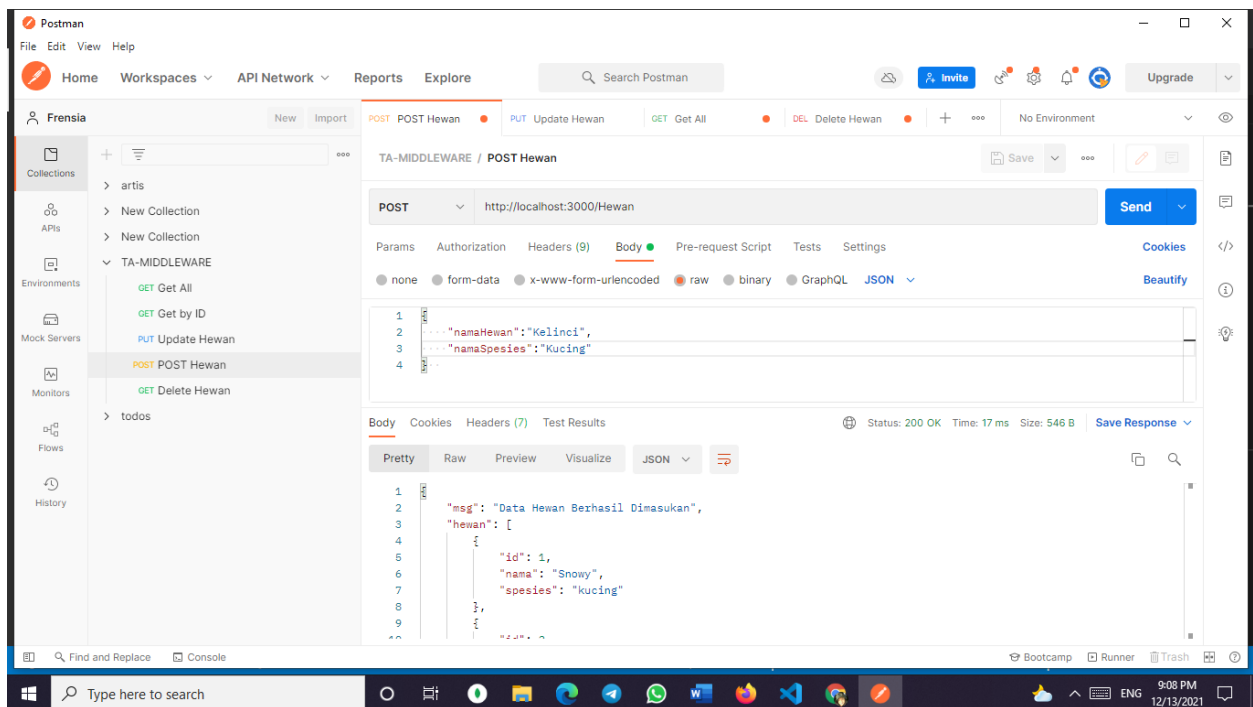


The screenshot shows the Visual Studio Code editor with the file `index.js` open. The code defines a `logger` middleware function and a `postChecker` function. The `logger` function logs the message "Middleware logger berjalan" and calls `next()`. The `postChecker` function defines a `payload` object. The terminal output shows the following messages:

```
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Example app listening at http://localhost:3000
Middleware logger berjalan
```

-Middleware postChecker

Jika data yang telah diatur ada



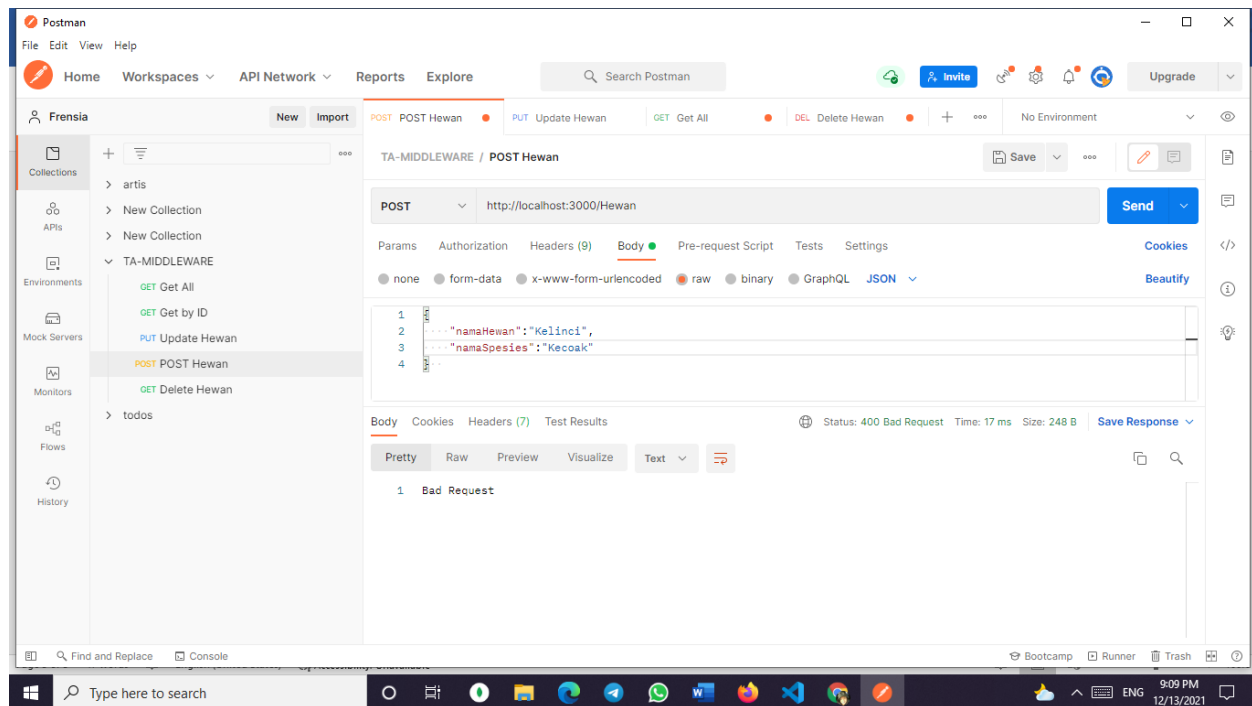
The screenshot shows the Postman application with a POST request to `http://localhost:3000/Hewan`. The request body is a JSON object with the following structure:

```
{
  "namaHewan": "Kelinci",
  "namaSpesies": "Kucing"
}
```

The response status is 200 OK, and the response body is a JSON object with the following structure:

```
{
  "msg": "Data Hewan Berhasil Dimasukan",
  "hewan": [
    {
      "id": 1,
      "nama": "Snowy",
      "spesies": "kucing"
    }
  ]
}
```

Jika data yang ada tidak sesuai aturan



Terimakasih.