

LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK



RIYADI TRI WALUYA BUDI

2308065

SIK A3

UNIVERSITAS PENDIDIKAN INDONESIA

SISTEM INFORMASI KELAUTAN

2024

I.PENDAHULUAN

Di era digital saat ini, keterampilan dalam pengembangan aplikasi web menjadi sangat penting. Salah satu aspek utama dalam pengembangan adalah kemampuan untuk melakukan operasi dasar pada data, yang dikenal sebagai CRUD (Create, Read, Update, Delete). Praktik ini bertujuan memberikan pemahaman mendalam tentang implementasi CRUD menggunakan Node.js sebagai backend dan MySQL sebagai database. Node.js adalah platform yang memungkinkan pengembang menjalankan JavaScript di sisi server, menawarkan kecepatan dan efisiensi dalam pengolahan data. Sementara itu, MySQL adalah sistem manajemen basis data relasional yang populer karena stabilitas dan kemudahan penggunaannya.

Tujuan dari praktik ini adalah:

1. Memahami konsep dasar CRUD.
2. Mempelajari cara menghubungkan Node.js dengan MySQL.
3. Membangun aplikasi sederhana yang dapat melakukan operasi CRUD pada data.

Cakupan praktik ini mencakup:

- Menginstal dan mengkonfigurasi Node.js dan MySQL.
- Membuat database dan tabel di MySQL.
- Melakukan operasi create, read, update, dan delete dengan Node.js.
- Pengujian aplikasi serta penanganan kesalahan.

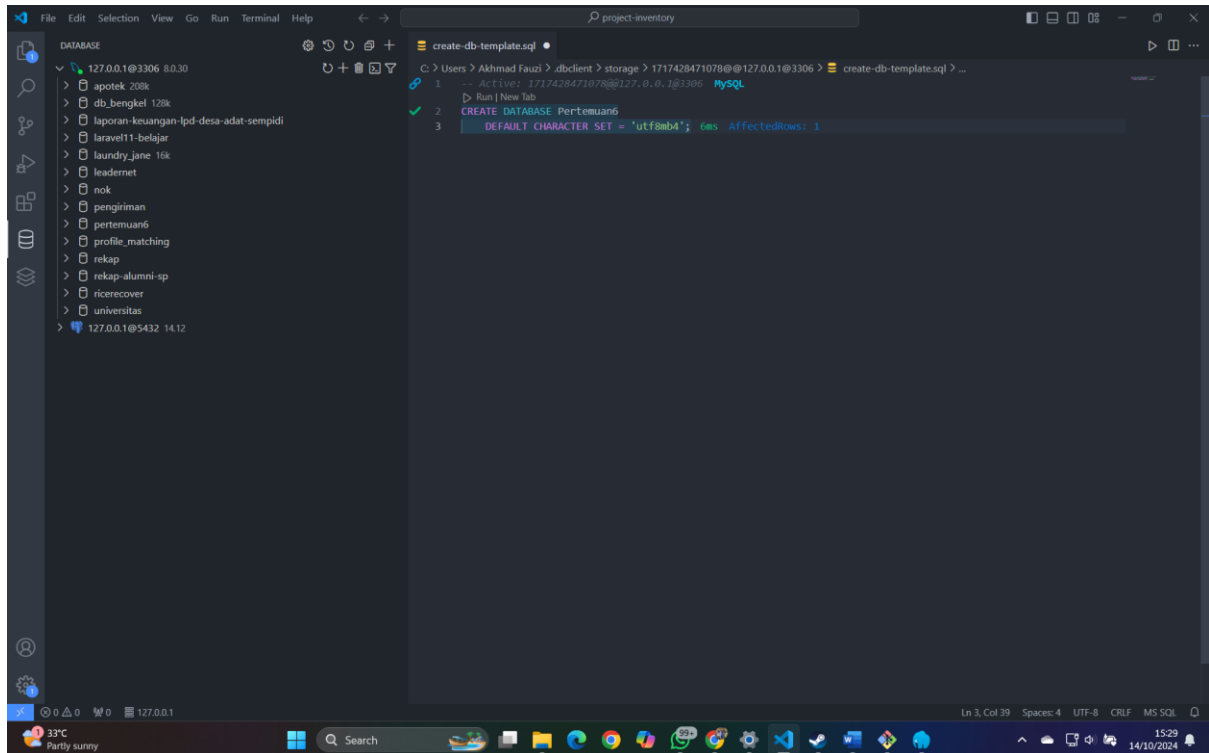
Dengan memahami dan menguasai praktik-praktik ini, peserta akan dapat mengembangkan aplikasi web yang interaktif dan responsif serta efektif dalam mengelola data.

II.ALAT DAN BAHAN

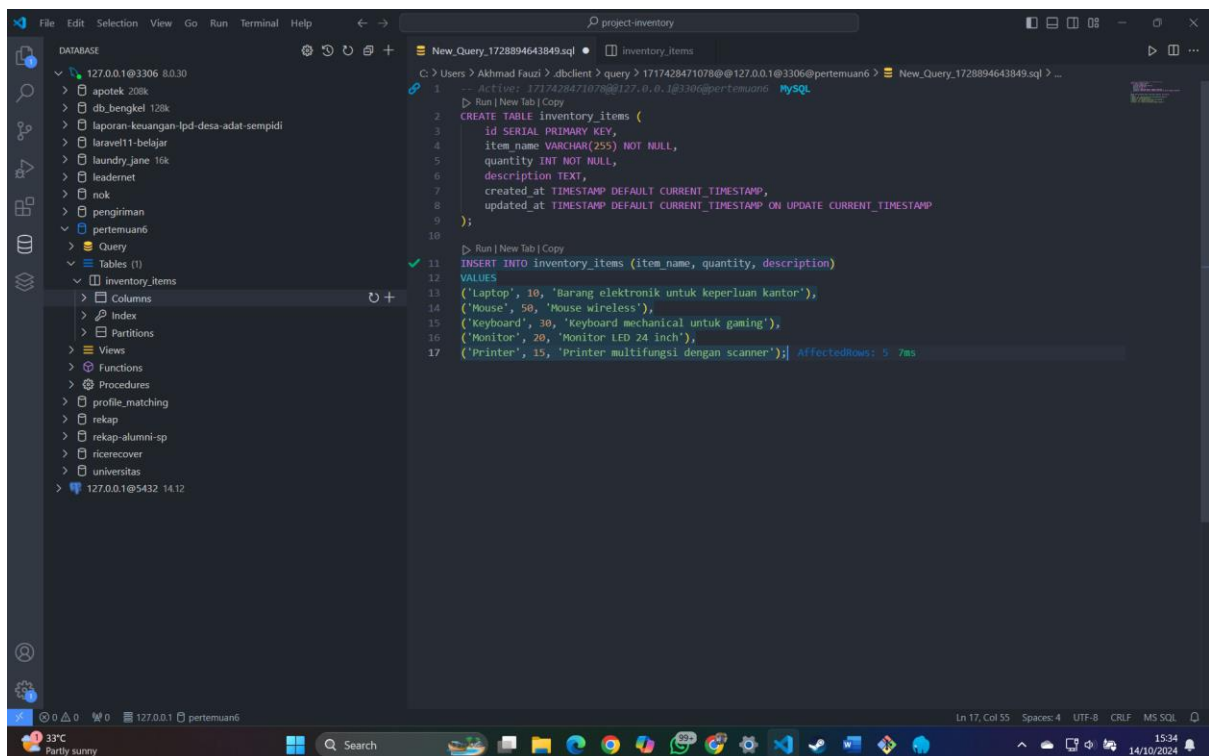
- VScode
- Node.js
- MySQL

III.LANGKAH – LANGKAH

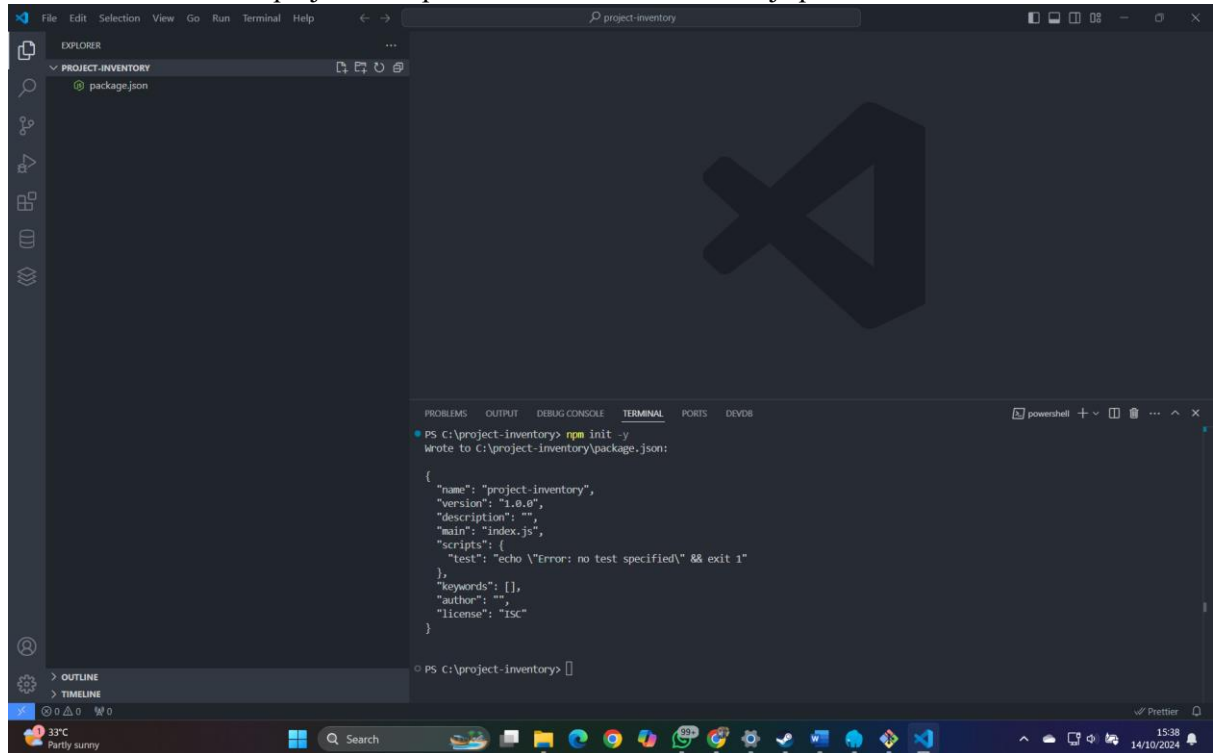
1. Pertama, aktifkan mysql di xampp kemudian connect ke database mysql menggunakan vscode. Setelah itu buat database baru dengan nama “Pertemuan 6”.



2. Setelah itu buat table baru dengan nama “inventory” dan sekaligus di isi data dummy



3. Kemudian buat folder project baru pada vscode lalu insiasi node.js pada terminal vscode

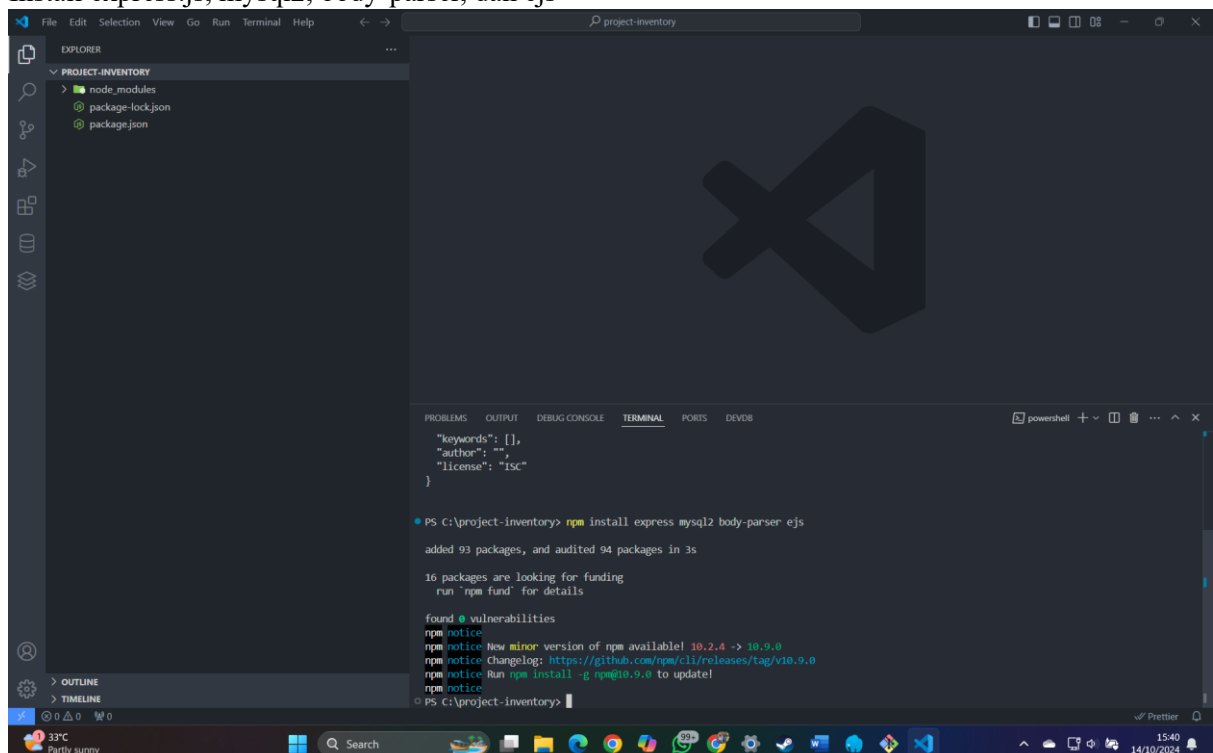


The screenshot shows the VS Code interface with a new project named 'project-inventory'. The Explorer sidebar on the left shows a folder named 'PROJECT-INVENTORY' containing a file named 'package.json'. The main editor area is empty, displaying the VS Code logo. The Terminal window at the bottom shows the command 'npm init -y' being executed, which has created the 'package.json' file. The output of the command is displayed in the terminal, showing the contents of the newly created 'package.json' file.

```
PS C:\project-inventory> npm init -y
wrote to C:\project-inventory\package.json:

{
  "name": "project-inventory",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

4. Install express.js, mysql2, body-parser, dan ejs



The screenshot shows the VS Code interface with the 'project-inventory' project. The Explorer sidebar on the left shows a folder named 'PROJECT-INVENTORY' containing files named 'package-lock.json' and 'package.json'. The main editor area is empty, displaying the VS Code logo. The Terminal window at the bottom shows the command 'npm install express mysql2 body-parser ejs' being executed. The output of the command is displayed in the terminal, showing the installation progress and the resulting package list.

```
PS C:\project-inventory> npm install express mysql2 body-parser ejs
added 93 packages, and audited 94 packages in 3s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New minor version of npm available! 10.2.4 -> 10.9.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.9.0
npm notice Run `npm install -g npm@10.9.0` to update!
npm notice
```

5. Buat file app.js kemudian buat source code untuk koneksi ke database kemudian testing run untuk mengecek apakah database sudah berhasil terkoneksi atau belum

The screenshot shows a Visual Studio Code editor with a project named 'project-inventory'. The Explorer sidebar on the left shows the file structure: 'PROJECT-INVENTORY' containing 'node_modules', 'app.js', 'package-lock.json', and 'package.json'. The main editor displays the 'app.js' file with the following code:

```
1 const express = require('express');
2 const mysql = require('mysql2');
3 const bodyParser = require('body-parser');
4
5 const app = express();
6 app.use(bodyParser.urlencoded({ extended: false }));
7 app.use(bodyParser.json());
8
9 const conn = mysql.createConnection({
10   host: 'localhost',
11   user: 'root',
12   password: '',
13   database: 'pertemuan6'
14 });
15
16 conn.connect((err) => {
17   if(err){
18     console.error("Error MySQL Connection:", err.stack);
19     return
20   }
21   console.log("Success MySQL Connection: " + conn.threadId);
22 });
```

The bottom panel shows the TERMINAL output:

```
16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New minor version of npm available! 10.2.4 -> 10.9.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.9.0
npm notice Run `npm install -g npm@10.9.0` to update!
PS C:\project-inventory> node app.js
Success MySQL Connection: 11
PS C:\project-inventory>
```

6. Jika sudah terkoneksi, Langkah selanjutnya adalah mencoba untuk membuat view. Tambahkan beberapa baris code untuk routes ke halaman awal dari web

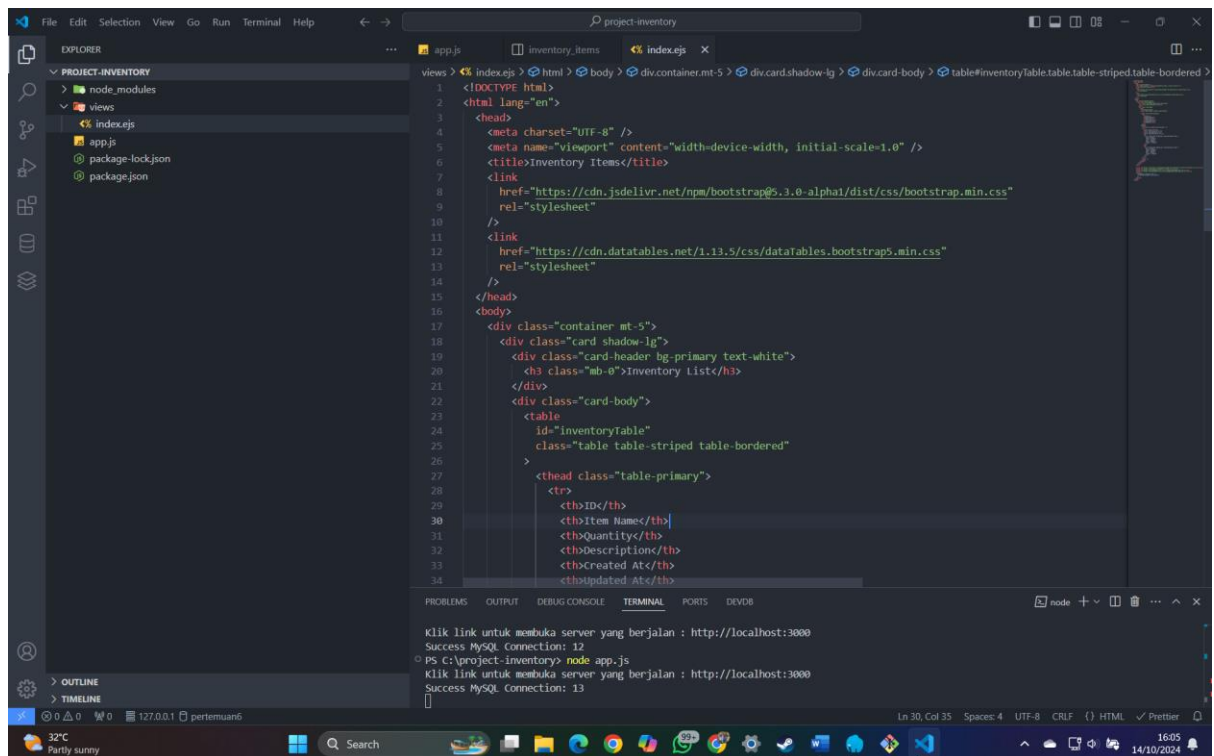
The screenshot shows the same Visual Studio Code editor with the 'app.js' file updated. The Explorer sidebar now includes a 'views' folder and an 'index.ejs' file. The 'app.js' code is updated as follows:

```
4
5 const app = express();
6 app.use(bodyParser.urlencoded({ extended: false }));
7 app.use(bodyParser.json());
8
9 const conn = mysql.createConnection({
10   host: 'localhost',
11   user: 'root',
12   password: '',
13   database: 'pertemuan6'
14 });
15
16 conn.connect((err) => {
17   if(err){
18     console.error("Error MySQL Connection:", err.stack);
19     return
20   }
21   console.log("Success MySQL Connection: " + conn.threadId);
22 });
23
24 app.set('view engine', 'ejs');
25
26 app.get('/', (req, res) => {
27   const query = "select * from inventory_items";
28   conn.query(query, (err, result) => {
29     res.render('index', {inventory : result});
30   });
31 });
32
33 app.listen(3000, () => {
34   console.log("Klik link untuk membuka server yang berjalan : http://localhost:3000");
35 });
```

The TERMINAL output shows the application running successfully:

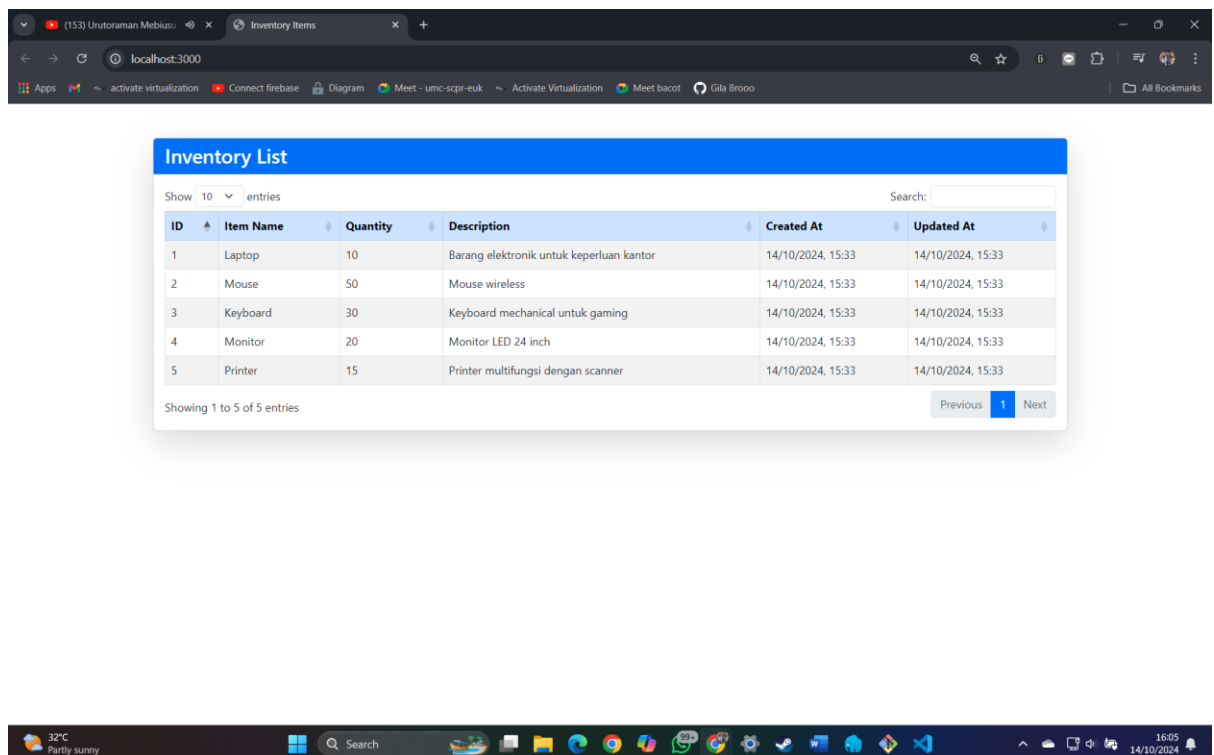
```
npm notice Run `npm install -g npm@10.9.0` to update!
npm notice
PS C:\project-inventory> node app.js
Success MySQL Connection: 11
PS C:\project-inventory>
```

7. Buat halaman view agar bisa di tampilkan pada browser

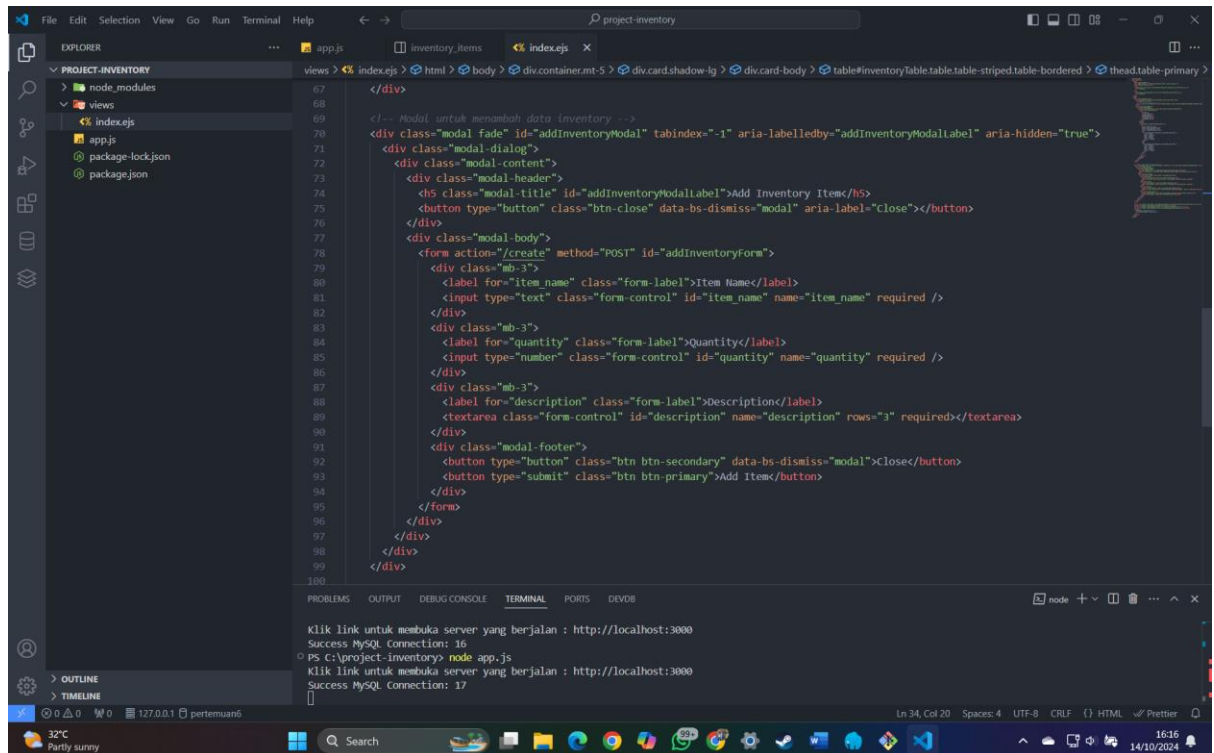


```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8" />
5   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6   <title>Inventory Items</title>
7   <link
8     href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"
9     rel="stylesheet"
10  />
11   <link
12     href="https://cdn.datatables.net/1.13.5/css/dataTables.bootstrap5.min.css"
13     rel="stylesheet"
14  />
15 </head>
16 <body>
17   <div class="container mt-5">
18     <div class="card shadow-lg">
19       <div class="card-header bg-primary text-white">
20         <h3 class="mb-0">Inventory List</h3>
21       </div>
22       <div class="card-body">
23         <table
24           id="inventoryTable"
25           class="table table-striped table-bordered"
26         >
27           <thead class="table-primary">
28             <tr>
29               <th>ID</th>
30               <th>Item Name</th>
31               <th>Quantity</th>
32               <th>Description</th>
33               <th>Created At</th>
34               <th>Updated At</th>
```

8. Berikut adalah hasil tampilan dari browsernya



9. Langkah selanjutnya membuat fungsi untuk create data inventory. Buat tombol untuk add data yang memanggil modal bootstrap.

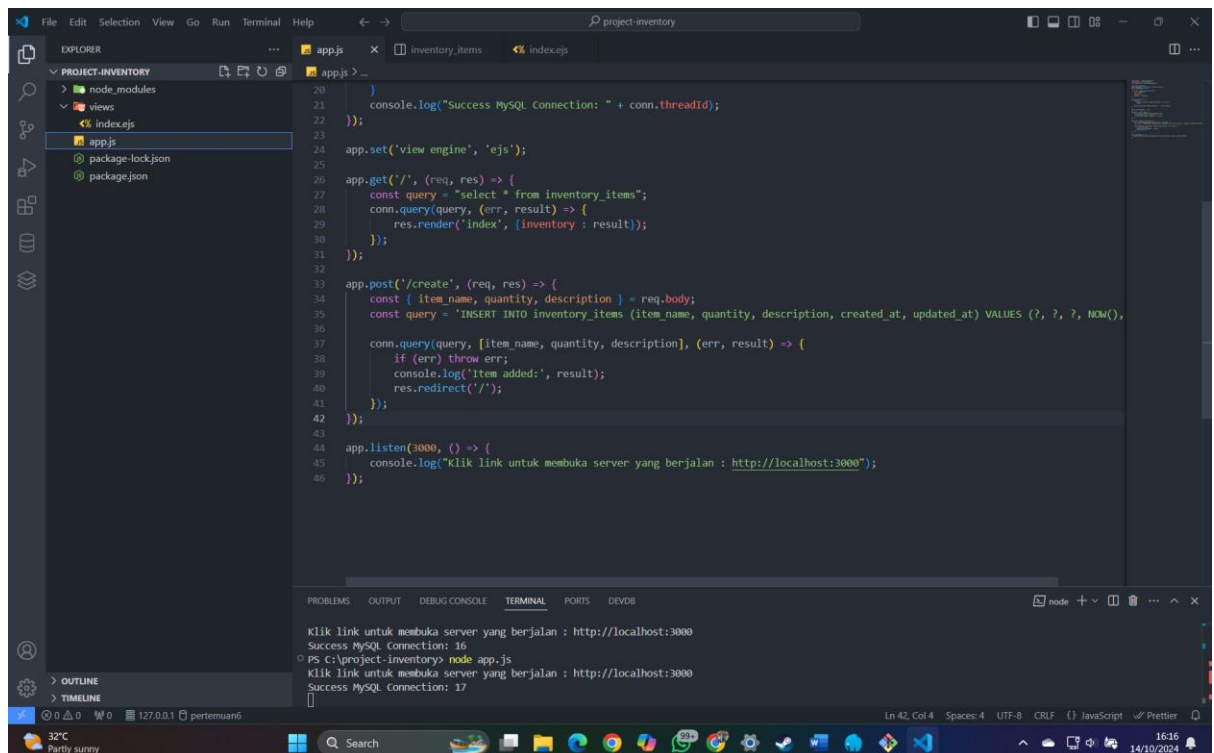


```
67 </div>
68
69 <!-- Modal untuk menambah data inventory -->
70 <div class="modal fade" id="addInventoryModal" tabindex="-1" aria-labelledby="addInventoryModalLabel" aria-hidden="true">
71   <div class="modal-dialog">
72     <div class="modal-content">
73       <div class="modal-header">
74         <h5 class="modal-title" id="addInventoryModalLabel">Add Inventory Item</h5>
75         <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
76       </div>
77       <div class="modal-body">
78         <form action="/create" method="POST" id="addInventoryForm">
79           <div class="mb-3">
80             <label for="item_name" class="form-label">Item Name</label>
81             <input type="text" class="form-control" id="item_name" name="item_name" required />
82           </div>
83           <div class="mb-3">
84             <label for="quantity" class="form-label">Quantity</label>
85             <input type="number" class="form-control" id="quantity" name="quantity" required />
86           </div>
87           <div class="mb-3">
88             <label for="description" class="form-label">Description</label>
89             <textarea class="form-control" id="description" name="description" rows="3" required></textarea>
90           </div>
91           <div class="modal-footer">
92             <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>
93             <button type="submit" class="btn btn-primary">Add Item</button>
94           </div>
95         </form>
96       </div>
97     </div>
98   </div>
99 </div>
100
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS DEVICE

Klik link untuk membuka server yang berjalan : <http://localhost:3000>
Success MySQL Connection: 16
PS C:\project-inventory> node app.js
Klik link untuk membuka server yang berjalan : <http://localhost:3000>
Success MySQL Connection: 17

10. Buat juga routes untuk menangani create data inventorynya

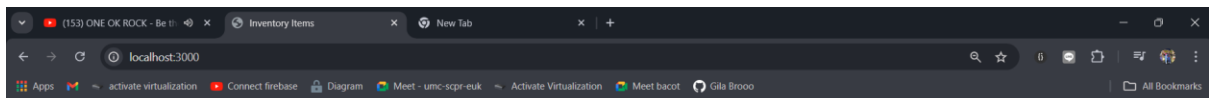
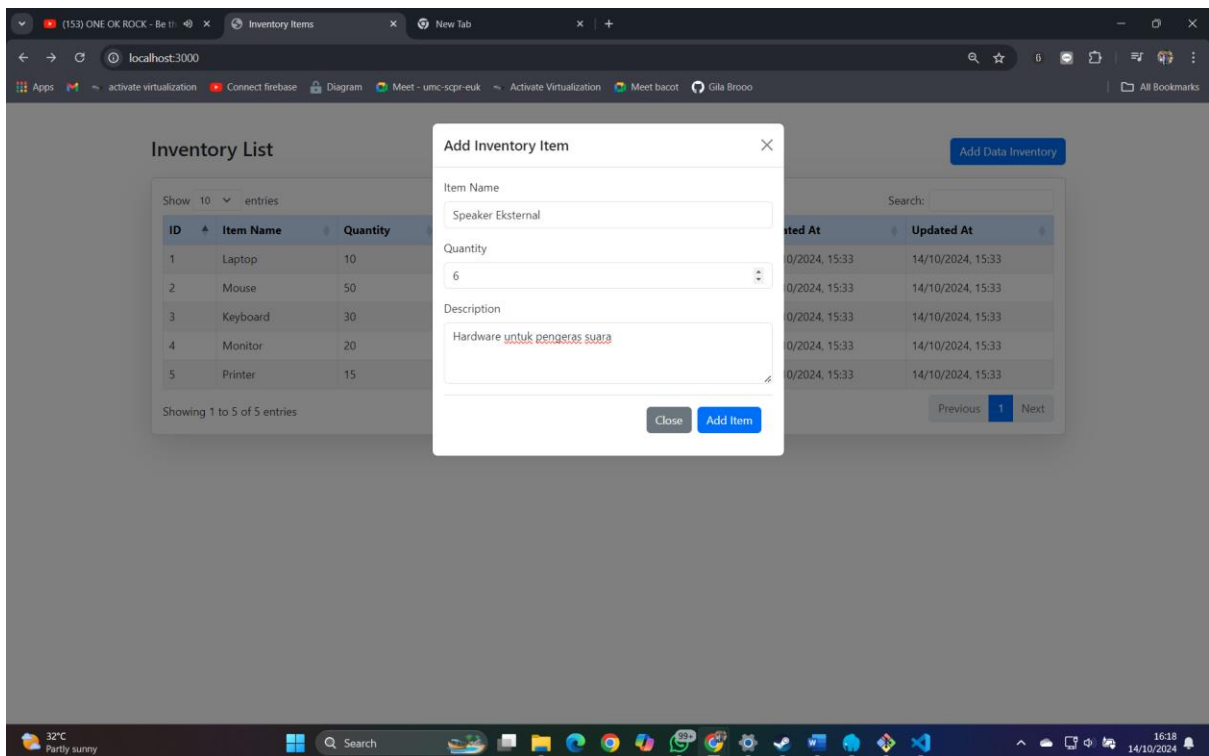


```
20 }
21 console.log("Success MySQL Connection: " + conn.threadId);
22 });
23
24 app.set('view engine', 'ejs');
25
26 app.get('/', (req, res) => {
27   const query = "select * from inventory_items";
28   conn.query(query, (err, result) => {
29     res.render('index', {inventory : result});
30   });
31 });
32
33 app.post('/create', (req, res) => {
34   const { item_name, quantity, description } = req.body;
35   const query = "INSERT INTO inventory_items (item_name, quantity, description, created_at, updated_at) VALUES (?, ?, ?, NOW(), NOW())";
36   conn.query(query, [item_name, quantity, description], (err, result) => {
37     if (err) throw err;
38     console.log("Item added:", result);
39     res.redirect('/');
40   });
41 });
42
43
44 app.listen(3000, () => {
45   console.log("Klik link untuk membuka server yang berjalan : http://localhost:3000");
46 });
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS DEVICE

Klik link untuk membuka server yang berjalan : <http://localhost:3000>
Success MySQL Connection: 16
PS C:\project-inventory> node app.js
Klik link untuk membuka server yang berjalan : <http://localhost:3000>
Success MySQL Connection: 17

11. Berikut hasilnya Ketika di coba untuk tambah data



Inventory List

Show: 10 entries

Search:

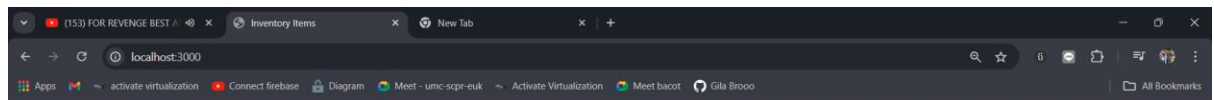
ID	Item Name	Quantity	Description	Created At	Updated At
1	Laptop	10	Barang elektronik untuk keperluan kantor	14/10/2024, 15:33	14/10/2024, 15:33
2	Mouse	50	Mouse wireless	14/10/2024, 15:33	14/10/2024, 15:33
3	Keyboard	30	Keyboard mechanical untuk gaming	14/10/2024, 15:33	14/10/2024, 15:33
4	Monitor	20	Monitor LED 24 inch	14/10/2024, 15:33	14/10/2024, 15:33
5	Printer	15	Printer multifungsi dengan scanner	14/10/2024, 15:33	14/10/2024, 15:33
8	Speaker Eksternal	6	Hardware untuk pengeras suara	14/10/2024, 16:18	14/10/2024, 16:18

Showing 1 to 6 of 6 entries

Previous 1 Next



12. Selanjutnya membuat fungsi untuk update data. Buat kolom action untuk menambahkan button action di tiap datanya. Terdapat 2 action yaitu update dan delete. Kemudian buat juga modal untuk melakukan update dan delete datanya



Inventory List

Add Data Inventory

Show 10 entries

Search:

ID	Item Name	Quantity	Description	Created At	Updated At	Actions
1	Laptop	10	Barang elektronik untuk keperluan kantor	14/10/2024, 15:33	14/10/2024, 15:33	<button>Update</button> <button>Delete</button>
2	Mouse	50	Mouse wireless	14/10/2024, 15:33	14/10/2024, 15:33	<button>Update</button> <button>Delete</button>
3	Keyboard	30	Keyboard mechanical untuk gaming	14/10/2024, 15:33	14/10/2024, 15:33	<button>Update</button> <button>Delete</button>
4	Monitor	20	Monitor LED 24 inch	14/10/2024, 15:33	14/10/2024, 15:33	<button>Update</button> <button>Delete</button>
5	Printer	15	Printer multifungsi dengan scanner	14/10/2024, 15:33	14/10/2024, 15:33	<button>Update</button> <button>Delete</button>
8	Speaker Eksternal	6	Hardware untuk pengeras suara	14/10/2024, 16:18	14/10/2024, 16:18	<button>Update</button> <button>Delete</button>

Showing 1 to 6 of 6 entries

Previous 1 Next



14. Kemudian buat routes untuk menangani permintaan update dan delete datanya.

```

File Edit Selection View Go Run Terminal Help
project-inventory

EXPLORER
PROJECT-INVENTORY
  node_modules
  views
  index.js
  app.js
  package-lock.json
  package.json

app.js
39 console.log("Item added:", result);
40 res.redirect('/');
41 });
42 });
43
44 // Route untuk update data inventory
45 app.post('/update', (req, res) => {
46   const { id, item_name, quantity, description } = req.body;
47   const query = 'UPDATE Inventory_items SET item_name = ?, quantity = ?, description = ?, updated_at = NOW() WHERE id = ?';
48
49   conn.query(query, [item_name, quantity, description, id], (err, result) => {
50     if (err) throw err;
51     console.log("Item updated:", result);
52     res.redirect('/');
53   });
54 });
55
56 // Route untuk menghapus data inventory
57 app.post('/delete', (req, res) => {
58   const { id } = req.body;
59   const query = 'DELETE FROM Inventory_items WHERE id = ?';
60
61   conn.query(query, [id], (err, result) => {
62     if (err) throw err;
63     console.log("Item deleted:", result);
64     res.redirect('/');
65   });
66 });
67
68 app.listen(3000, () => {
69   console.log("Klik link untuk membuka server yang berjalan : http://localhost:3000");
70 });

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS DEVICE
affectedRows: 1,
insertId: 0,
info: '',
serverStatus: 2,
warningStatus: 0,
changedRows: 0
}
Item updated: ResultSetHeader {
  fieldCount: 0,
  affectedRows: 1,
  insertId: 0,
  info: 'Rows matched: 1 Changed: 1 Warnings: 0',
  serverStatus: 2,
  warningStatus: 0,
  changedRows: 1
}
node powershell
32°C Partly sunny
16:29 14/10/2024

```

15. Berikut contoh Ketika melakukan perubahan data

Data sebelum di lakukan perubahan

Inventory List

Add Data Inventory

Show

10

 entries

Search:

ID	Item Name	Quantity	Description	Created At	Updated At	Actions
1	Laptop	10	Barang elektronik untuk keperluan kantor	14/10/2024, 15:33	14/10/2024, 15:33	<div>UpdateDelete</div>
2	Mouse	50	Mouse wireless	14/10/2024, 15:33	14/10/2024, 15:33	<div>UpdateDelete</div>
3	Keyboard	30	Keyboard mechanical untuk gaming	14/10/2024, 15:33	14/10/2024, 15:33	<div>UpdateDelete</div>
4	Monitor	25	Monitor LED 24 inch	14/10/2024, 15:33	14/10/2024, 16:29	<div>UpdateDelete</div>

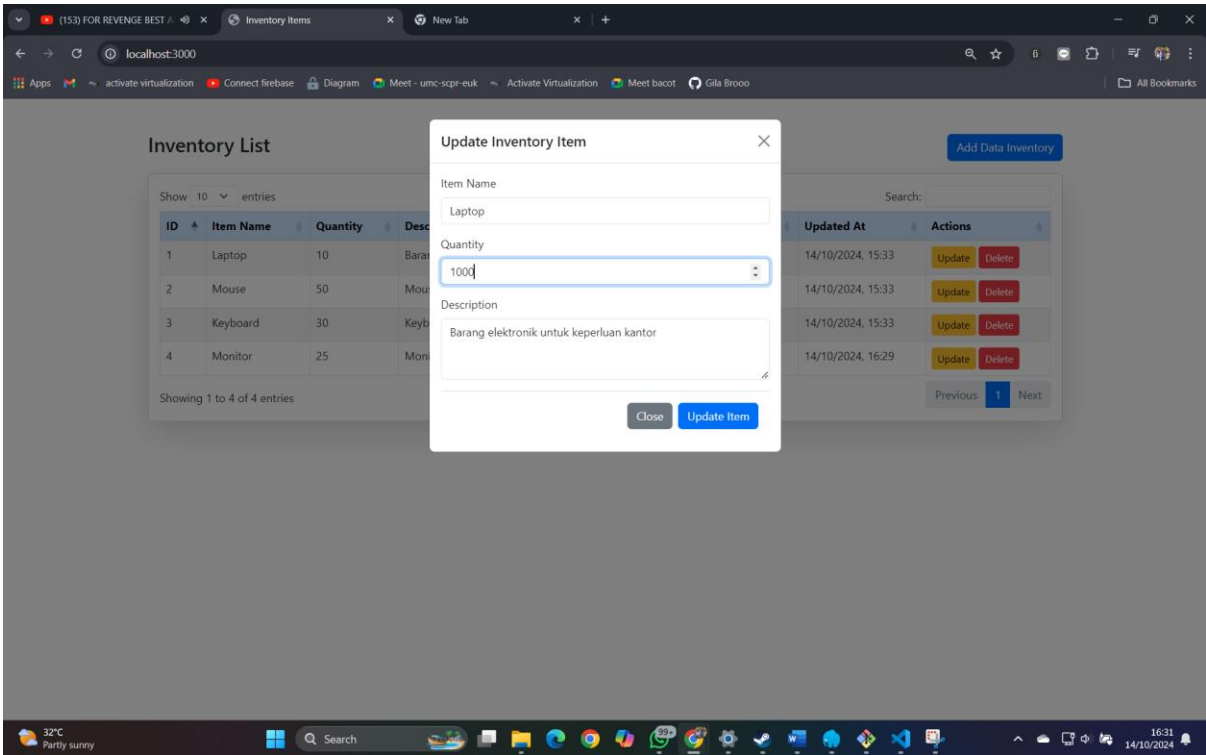
Showing 1 to 4 of 4 entries

Previous

1

Next

Update:



Hasil update :

Inventory List

Add Data Inventory

Show

10

 entries

Search:

ID	Item Name	Quantity	Description	Created At	Updated At	Actions
1	Laptop	1000	Barang elektronik untuk keperluan kantor	14/10/2024, 15:33	14/10/2024, 16:31	<div>UpdateDelete</div>
2	Mouse	50	Mouse wireless	14/10/2024, 15:33	14/10/2024, 15:33	<div>UpdateDelete</div>
3	Keyboard	30	Keyboard mechanical untuk gaming	14/10/2024, 15:33	14/10/2024, 15:33	<div>UpdateDelete</div>
4	Monitor	25	Monitor LED 24 inch	14/10/2024, 15:33	14/10/2024, 16:29	<div>UpdateDelete</div>

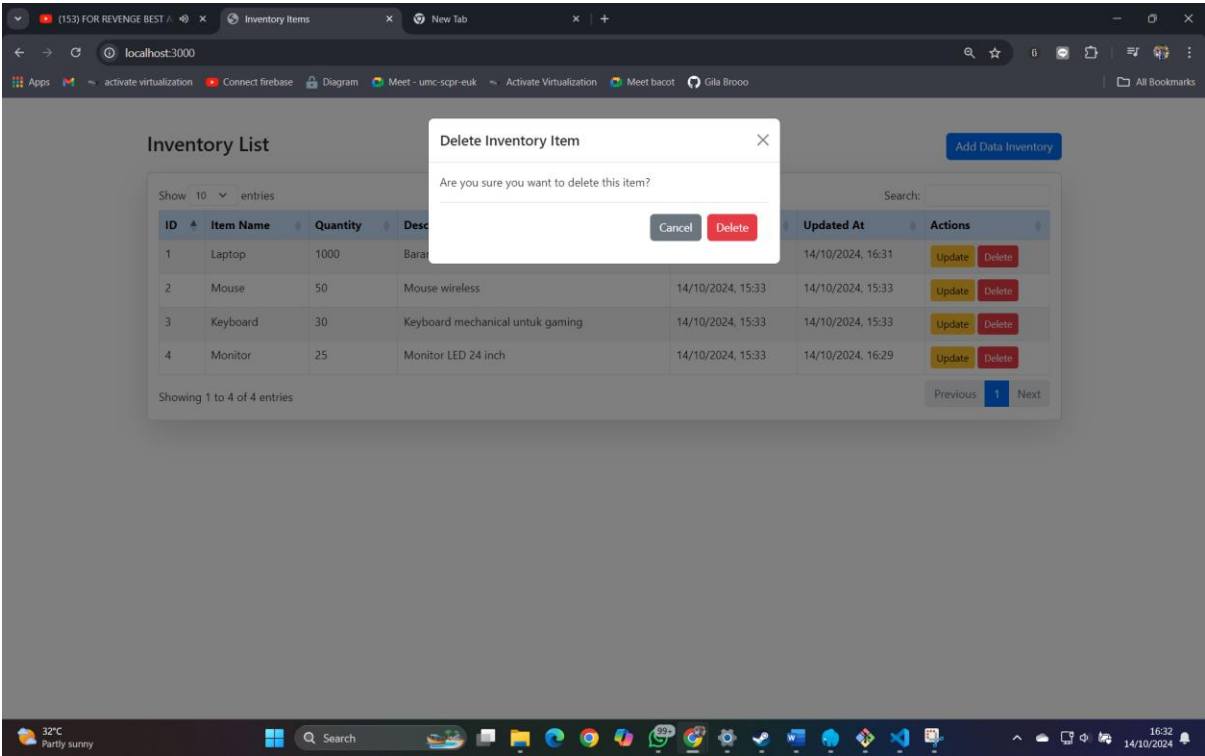
Showing 1 to 4 of 4 entries

Previous

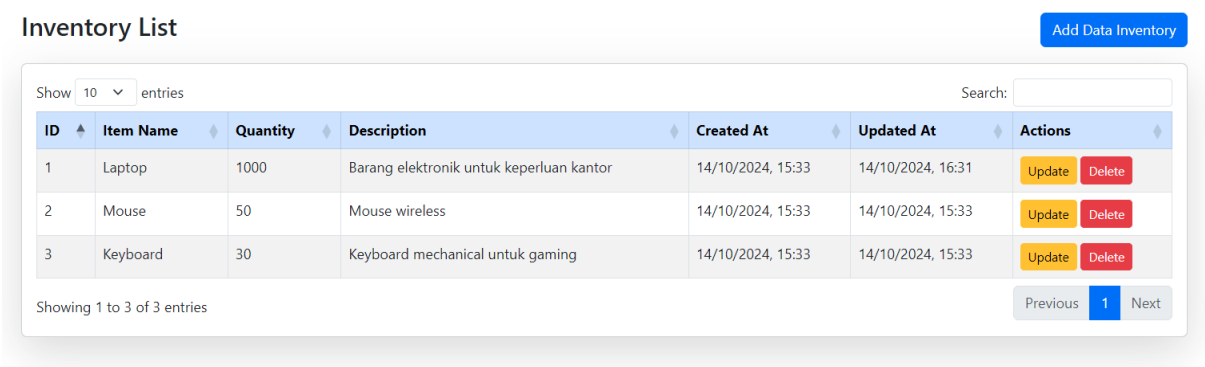
1

Next

Delete :



Hasil delete :



IV.KESIMPULAN

1. Fungsi CRUD: Praktik ini menunjukkan bagaimana sistem dapat Memproses data inventaris secara efisien melalui empat dasar elemen. operasi untuk membuat, membaca, memperbarui dan menghapus data.
- 2.Teknologi yang digunakan: Penggunaan bahasa pemrograman seperti PHP, JavaScript dan database seperti MySQL memungkinkan pengelolaan data yang dinamis dan interaktif.
- 3.Antarmuka Pengguna: Desain antarmuka yang sederhana dan intuitif membantu pengguna berinteraksi dengan sistem dengan lebih mudah, sehingga meningkatkan pengalaman pengguna.
- 4.Manajemen Data: Sistem ini mendukung pengelolaan data inventaris yang lebih baik, membantu melacak dan mengontrol inventaris.
- 5.Pentingnya Otentikasi: Praktik ini menekankan pentingnya validasi data untuk menjaga integritas dan konsistensi informasi dalam database.

Secara keseluruhan, Praktik ini membantu Anda lebih memahami pengembangan aplikasi web dan pentingnya sistem informasi dalam manajemen inventaris.