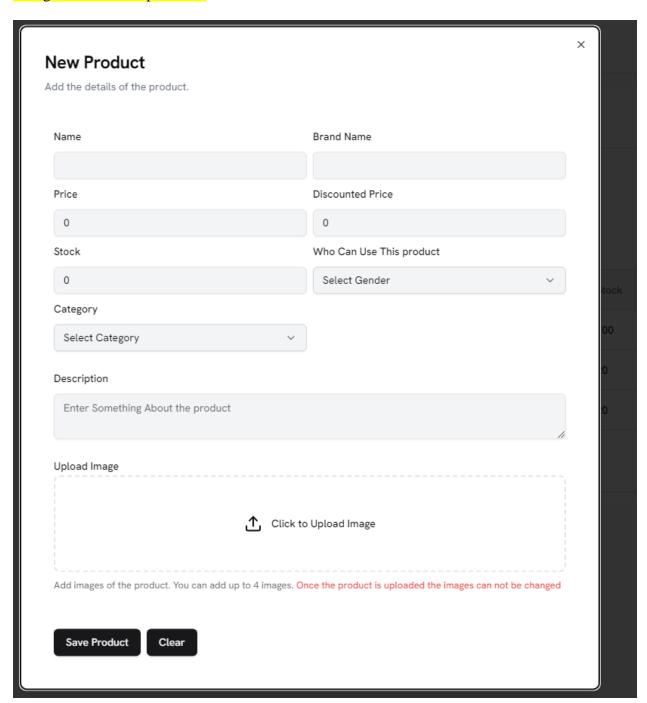


Testing

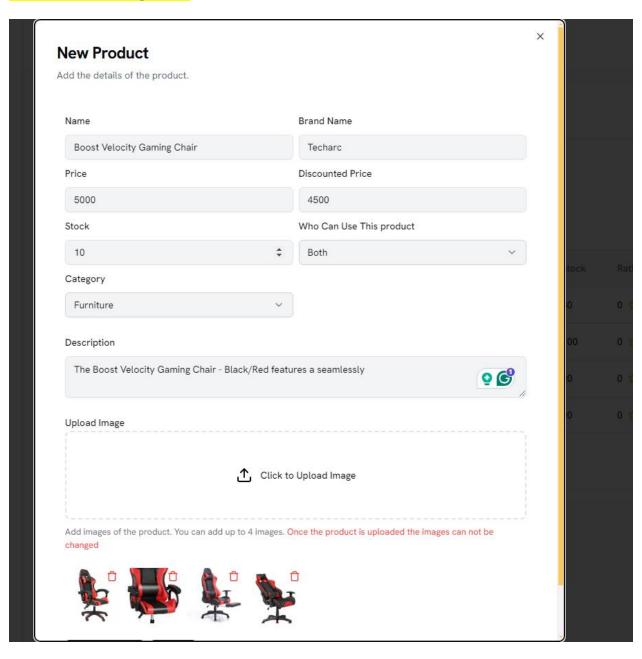
Using this UI to add products:



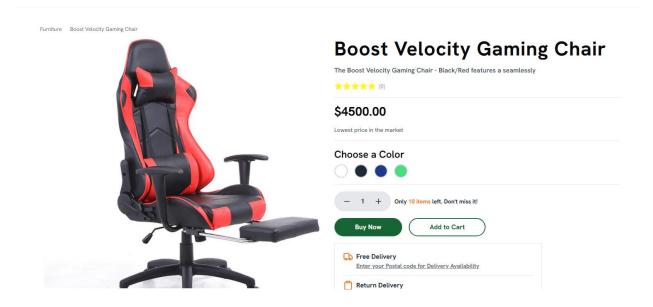
This is the code used to add the product, using prisma, sql, express and node is

```
const createProducts = async (req, res) => {
   try {
       const { images, ...productData } = req.body;
       if (!images || !Array.isArray(images) || images.length === 0) {
           return res.status(404).json({
               message: "Images not found or invalid format",
           });
       }
       // Create the product first
       const product = await prisma.product.create({
           data: {
               ...productData,
           },
       });
       // Create images linked to the product
       const imagePromises = images.map((img) =>
           prisma.image.create({
               data: {
                   url: img,
                   altText: img.altText | null,
                   productId: product.id,
               },
           })
       );
       const allImages = await Promise.all(imagePromises);
       return res.status(200).json({
           message: "Product created successfully",
           product,
           images: allImages,
       });
   } catch (error) {
       console.error("Error while creating the product:", error.message);
       return res.status(500).json({
           message: "Error while creating the product",
       });
```

Lets Add A Gaming Chair



Added this



SQL Queries:

1. Select all users

SELECT * FROM User;



2. Select specific columns from the Product table

SELECT id, name, price FROM Product;

3. Select products where stock is less than 10

Select name, stock from product where stock < 10;

```
MariaDB [shope_ease]> SELECT name, stock FROM Product WHERE stock < 10;
Empty set (0.000 sec)
```

4. Select users from a specific city

SELECT name, email FROM User WHERE city = 'isb';

Projection Queries

5. Select distinct countries of users

SELECT DISTINCT country FROM User;

```
MariaDB [shope_ease]> SELECT DISTINCT country FROM User;

+-----+
| country |

+-----+
| pakistan |

+-----+
1 row in set (0.000 sec)
```

6. Select names of users and their cities

SELECT name, city FROM User;

Joining Queries

7. Join User and CartItem to fetch user cart items

SELECT u.name, c.productId, c.quantity

FROM User u

JOIN CartItem c ON u.id = c.userId;

8. Join Product and Image to fetch product images

SELECT p.name AS ProductName, i.url AS ImageURL

FROM Product p

JOIN Image i ON p.id = i.productId;

```
ariaDB [shope_ease]> SELECT p.name AS ProductName, i.url AS ImageURL
   -> FROM Product p
-> JOIN Image i ON p.id = i.productId;
Armor Hoodie Black
                                              uploads/files/1736063775056_1000084-ArmorHoodieBlack_01_bf22178f-324e-437f-af68-be4edd1f4a29.webp
                                             Armor Hoodie Black
Basic Hoodie - 5001
Basic Hoodie - 5001
Basic Hoodie - 5001
Basic Hoodie - 5001
Apple iPhone 16 Pro Max 256GB
SL A3368 iREST MASSAGE CHAIR
                                               uploads/files/1736593266456_ntc-Hero-Line-Up.avif uploads/files/1736599800468_main3368.webp
                                               uploads/files/1736599826834_fourth.webp
uploads/files/1736599805907_3368_db4a8a54-a8b1-446c-984b-27c55b02c56c.webp
uploads/files/1736599793044_sla-235-3.webp
SL A3368 iREST MASSAGE CHAIR
SL A3368 iREST MASSAGE CHAIR
SL A3368 iREST MASSAGE CHAIR
Boost Velocity Gaming Chair
Boost Velocity Gaming Chair
Boost Velocity Gaming Chair
Boost Velocity Gaming Chair
                                               uploads/files/1736600443646_images (1).jfif uploads/files/1736600441340_images.jfif
                                              uploads/files/1736600447575_83c391dc3005442f1a1221846eb28059_800x.webp
uploads/files/1736600445690_1664123364_12_4159.jpg
```

9. Join CartItem, Product, and User to fetch cart details

SELECT u.name AS UserName, p.name AS ProductName, c.quantity, c.isOrdered

FROM CartItem c

JOIN Product p ON c.productId = p.id

JOIN User u ON c.userId = u.id;

```
MariaDB [shope_ease]> SELECT u.name AS UserName, p.name AS ProductName, c.quantity, c.isOrdered
    -> FROM CartItem c
    -> JOIN Product p ON c.productId = p.id
    -> JOIN User u ON c.userId = u.id;
                                              | quantity | isOrdered |
 UserName | ProductName
 ali akbar | Armor Hoodie Black
ali akbar | Basic Hoodie - 5001
                                                       1 |
 ali akbar | Armor Hoodie Black
                                                        5 |
                                                                    1 |
 ali akbar | Armor Hoodie Black
                                                        2 |
 ali akbar | Basic Hoodie - 5001
 ali akbar | Apple iPhone 16 Pro Max 256GB |
                                                                    0
 rows in set (0.001 sec)
```

Aggregation Queries

10. Count the total number of users

SELECT COUNT(*) AS TotalUsers FROM User;

11. Find the average price of products

SELECT AVG(price) AS AveragePrice FROM Product;

```
MariaDB [shope_ease]> SELECT AVG(price) AS AveragePrice FROM Product;

+-----+
| AveragePrice |
+-----+
| 2360 |
+-----+
1 row in set (0.000 sec)
```

12. Find the total revenue from all orders

SELECT SUM(totalAmount) AS TotalRevenue FROM `Order`;

```
MariaDB [shope_ease]> SELECT SUM(totalAmount) AS TotalRevenue FROM `Order`;

+------+
| TotalRevenue |

+------+
| 5630 |

+-----+
1 row in set (0.000 sec)
```

13. Find the maximum rating of products

SELECT MAX(rating) AS MaxRating FROM Product;

```
MariaDB [shope_ease]> SELECT MAX(rating) AS MaxRating FROM Product;
+-----+
| MaxRating |
+-----+
| 0 |
+-----+
1 row in set (0.001 sec)
```

14. Find the number of products in each category

SELECT category, COUNT(*) AS ProductCount

FROM Product

GROUP BY category;

Subqueries

15. Find users who have placed an order

SELECT name, email

FROM User

WHERE id IN (SELECT userId FROM `Order`);

16. Find products that are in a user's cart but not yet ordered

SELECT name

FROM Product

WHERE id IN (SELECT productId FROM CartItem WHERE isOrdered = FALSE);

17. Find the total revenue of a specific user

SELECT SUM(o.totalAmount) AS UserRevenue

FROM `Order` o

WHERE o.userId = (SELECT id FROM User WHERE name = 'Ali Akbar');

CRUD Operations

<u>Create:</u>

18.Insert a new user

INSERT INTO User (name, email, authOld, city, country, phone, createdAt)

VALUES ('Ali Akbar', 'ali@example.com', 'auth0|12345', 'Karachi', 'Pakistan', '1234567890', NOW());

```
MariaDB [shope_ease]> INSERT INTO User (name, email, auth0Id, city, country, phone, createdAt)
-> VALUES ('Ali Akbar', 'ali@example.com', 'auth0|12345', 'Karachi', 'Pakistan', '1234567890', NOW());
Query OK, 1 row affected (0.015 sec)
```

19.Insert a new product

INSERT INTO Product (name, brand, price, stock, category, gender, createdAt)

VALUES ('Smartphone', 'BrandX', 25000, 50, 'Electronics', 'BOTH', NOW());

```
MariaDB [shope_ease]> INSERT INTO Product (name, brand, price, stock, category, gender, createdAt)
-> VALUES ('Smartphone', 'BrandX', 25000, 50, 'Electronics', 'BOTH', NOW());
Query OK, 1 row affected, 1 warning (0.004 sec)
```

Read:

20. Retrieve products above a specific price

SELECT name, price FROM Product WHERE price > 20000;

21. Fetch all users who have more than one cart item

```
SELECT u.name, COUNT(c.id) AS CartItems
```

FROM User u

JOIN CartItem c ON u.id = c.userId

GROUP BY u.id

HAVING COUNT(c.id) > 1;

Update:

22. Update stock of a specific product

```
UPDATE Product
```

SET stock = stock + 10

WHERE id = 1;

```
MariaDB [shope_ease]> UPDATE Product
-> SET stock = stock + 10
-> WHERE id = 1;
Query OK, 0 rows affected (0.000 sec)
Rows matched: 0 Changed: 0 Warnings: 0
```

23.Mark a cart item as ordered

UPDATE CartItem

SET is Ordered = TRUE

WHERE id = 3;

```
MariaDB [shope_ease]> UPDATE CartItem
-> SET isOrdered = TRUE
-> WHERE id = 3;
Query OK, 0 rows affected (0.001 sec)
Rows matched: 0 Changed: 0 Warnings: 0
```

Delete:

24.Delete User

DELETE FROM User WHERE id = 5;

MariaDB [shope_ease]> DELETE FROM User WHERE id = 5; Query OK, 0 rows affected (0.001 sec)

25. Delete Product

Delete from product where id = 100;

MariaDB [shope_ease]> delete from product where id = 100; Query OK, 0 rows affected (0.000 sec)