LARAVEL

**SISTEM INFORMASI INVENTORY KONSENTRASI KEAHLIAN SIJA**

# Latar Belakang

Dalam era di mana informasi menjadi salah satu aset terpenting bagi berbagai institusi, sistem informasi telah menjadi tulang punggung yang mendukung efisiensi operasional, pengambilan keputusan yang tepat, dan pertumbuhan bisnis yang berkelanjutan. Konsep sistem informasi melibatkan integrasi teknologi, prosedur, dan sumber daya manusia untuk mengumpulkan, mengelola, dan mendistribusikan informasi yang diperlukan dalam lingkup organisasi. Hal ini meliputi infrastruktur teknologi seperti perangkat keras, perangkat lunak, jaringan, serta basis data, yang bekerja bersama-sama untuk mendukung proses operasional. Pentingnya sistem informasi terletak pada kemampuannya untuk meningkatkan efisiensi, membantu pengambilan keputusan yang lebih terinformasi, mengembangkan bisnis, dan meningkatkan komunikasi di dalam institusi.

Namun, dampak sistem informasi tidak hanya terbatas pada aspek positif. Meskipun sistem informasi memberikan banyak manfaat, ada pula tantangan dan risiko yang perlu diatasi. Ketergantungan yang berlebihan pada sistem informasi dapat mengakibatkan kerentanan saat terjadi gangguan atau kegagalan sistem. Ancaman terhadap keamanan data juga merupakan isu penting yang harus diatasi dengan serius, mengingat tingginya risiko kebocoran data atau serangan keamanan yang dapat merugikan institusi secara keseluruhan. Selain itu, investasi yang diperlukan dalam pengembangan, pemeliharaan, dan peningkatan sistem informasi seringkali membutuhkan anggaran yang besar, menjadi pertimbangan penting dalam strategi penggunaannya.

Dengan memahami kedalaman manfaat serta risiko yang terkait dengan sistem informasi, institusi dapat merancang strategi yang matang dalam pemanfaatan teknologi ini. Penggunaan sistem informasi yang efektif memerlukan perencanaan yang cermat, pengelolaan risiko yang baik, serta kesadaran akan kebutuhan akan keamanan dan investasi yang tepat guna. Dengan demikian, sistem informasi dapat menjadi alat yang sangat berharga dalam mendukung pertumbuhan dan kelangsungan institusi di era modern yang dipenuhi dengan kompleksitas informasi.

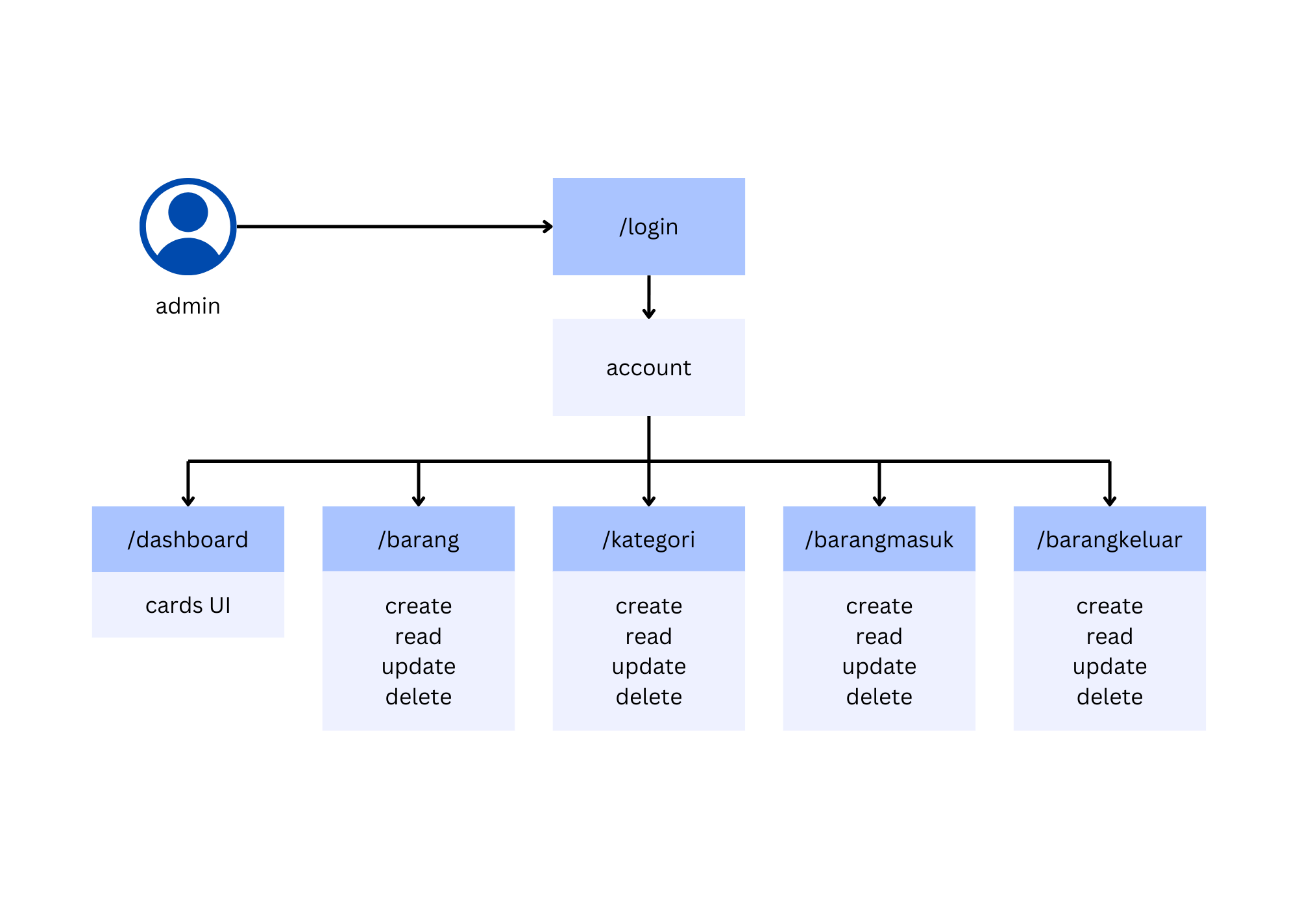
# Tujuan

Tujuan utama project ini adalah untuk menciptakan aplikasi inventori yang memungkinkan pengguna untuk mengelola dan melacak informasi terkait kategori, barang, serta transaksi barang masuk dan keluar. Integrasi yang baik antara bagian-bagian ini akan mempermudah pengelolaan stok dan transaksi. Dengan sistem yang terotomatisasi dan terstruktur, diharapkan dapat mengurangi waktu yang diperlukan dalam proses pencatatan, pembaruan data, dan pelacakan stok barang.

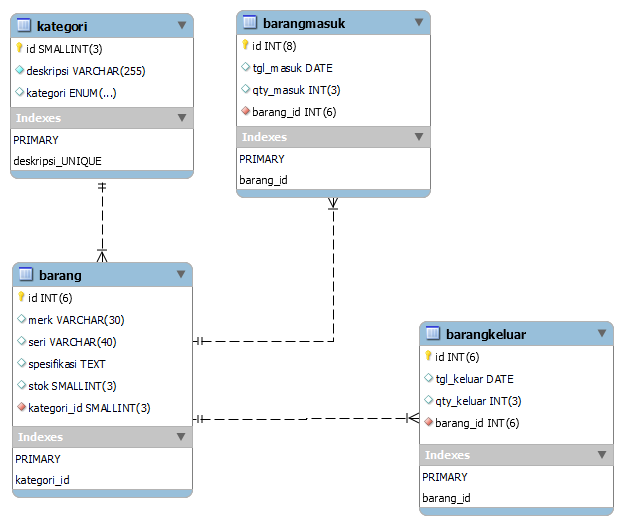
# Manfaat

Sistem inventory berbasis web dengan Laravel memungkinkan pengguna untuk dengan mudah dan cepat menambah, mengedit, serta menghapus data inventaris, mengurangi risiko kesalahan manusia. Dengan implementasi sistem CRUD, keakuratan dan integritas data persediaan terjaga, meningkatkan efisiensi pengelolaan inventaris secara menyeluruh.

# Desain Aplikasi



# Skema Database



# Skenario

Dalam rangka mencapai kualifikasi sebagai *web programmer* anda diminta untuk menyelesaikan tugas membuat aplikasi sistem inventory sekolah berbasis web menggunakan *framework* Laravel dengan ketentuan:

1. Sistem operasi : Linux Debian 12
   1. Hostname : daudf
   2. ServerName : jagokoding.test
   3. DocumentRoot : /var/www/jagokoding
2. Layanan Server
   1. *Web Server* : apache 2.4.56
   2. *Database Server* : mariadb
   3. *Language Server Side* : PHP 8.2.10
3. *Software Tools* : composer
4. PHP *framework* : laravel 10.26.2
5. *Project Name* : jagokoding
6. *Database Account*
   1. *Database Username* : jagokoding
   2. *Database Name* : jagkokding
   3. *Database Password* : 123
   4. *Database Hostname* : localhost
   5. *Database User Privileges* : All Privileges on jagokoding.\*
7. *Web Admin Template* : Bootstrap SB Admin 2

<https://startbootstrap.com/theme/sb-admin-2>

# Langkah Kerja

# Mengkonfigurasi Database

* Nama database : jagokoding
* User database : jagokoding
* Password database : 123

1. Menginstall mariadb

⇒ apt install mariadb-server -y

1. Membuat Database

⇒ create database jagokoding;

1. Membuat user

⇒ create user ‘jagokoding@’localhost’ identified by ‘123’;

1. Memberikan hak akses untuk user ke databse

⇒ grant all privileges on jagokoding.\* to ‘jagokoding@’localhost’ identified by ‘123’;

1. Refresh hak akses

⇒ flush privileges;

# Membuat Project Laravel

* Nama project : jagokoding
* Folder : /var/www/jagokoding

1. Menginstall Apache dan PHP
   1. Apache2

⇒ apt install apache2 -y

* 1. PHP v.8.2

⇒ apt-get install php8.2 php8.2-common php8.2-curl libapache2-mod-php php8.2-imap php8.2-redis php8.2-cli php8.2-snmp php8.2-xml php8.2-zip php8.2-mbstring php8.2-gd php8.2-xml php8.2-mysql php-mbstring -y

1. Menginstall composer

* apt install curl php-cli php-mbstring git unzip
* curl -sS https://getcomposer.org/installer -o composer-setup.php
* HASH=`curl -sS https://composer.github.io/installer.sig`
* echo $HASH
* php -r "if (hash\_file('SHA384', 'composer-setup.php') === '$HASH') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP\_EOL;"
* sudo php composer-setup.php --install-dir=/usr/local/bin --filename=composer
* composer

1. Membuat project laravel pada direktori /var/www/

⇒ composer create-project laravel/laravel jagokoding --prefer-dist

1. Mengubah permission direktori dan file

⇒ chown -R www-data:www-data jagokoding/

1. Mengkonfigurasi virtualhost

⇒ /etc/apache2/sites-available/jagokoding.conf

<VirtualHost \*:80>

        ServerName jagokoding.test

        ServerAdmin webmaster@localhost

        DocumentRoot /var/www/jagokoding/public

        ErrorLog ${APACHE\_LOG\_DIR}/error-jagokoding.log

        CustomLog ${APACHE\_LOG\_DIR}/access-jagokoding.log combined

       <Directory /var/www/jagokoding>

                Options Indexes FollowSymLinks

                AllowOverride All

                Require all granted

       </Directory>

</VirtualHost>

1. Mengaktifkan virtualhost: jagokoding.test

⇒ a2ensite jagokoding.conf

1. Mengkonfigurasi file hosts

⇒ nano /etc/hosts

⇒ C:\Windows\System32\drivers\etc\hosts

192.168.126.80 jagokoding.test

# Konfigurasi Database Laravel

1. Konfigurasi database pada file /var/www/jagokoding/.env

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=jagokoding

DB\_USERNAME=jagokoding

DB\_PASSWORD=123

# Membuat Struktur table dengan Migration

1. Membuat migration
2. Table kategori

⇒ php artisan make:migration create\_kategori\_table --create=kategori

1. Table barang

⇒ php artisan make:migration create\_barang\_table --create=barang

1. Table barangmasuk

⇒ php artisan make:migration create\_barangmasuk\_table --create=barangmasuk

1. Table barangkeluar

⇒ php artisan make:migration create\_barangkeluar\_table --create=barangkeluar

1. Mengedit file migration

Kode program :

<https://github.com/daudfirmann/inventory-projectlaravel/tree/main/database/migrations>

# Membuat Model

1. Membuat model
2. Model Kategori

⇒ php artisan make:model Kategori

1. Model Barang

⇒ php artisan make:model Barang

1. Model Barangmasuk

⇒ php artisan make:model Barangmasuk

1. Model Barangkeluar

⇒ php artisan make:model Barangkeluar

1. Mengedit file Model

Kode program :

<https://github.com/daudfirmann/inventory-projectlaravel/tree/main/app/Models>

# Membuat Controller

1. Membuat controller
2. Controller kategori

⇒ php artisan make:controller KategoriController

1. Controller barang

⇒ php artisan make:controller BarangController

1. Controller barangmasuk

⇒ php artisan make:controller BarangmasukController

1. Controller barangkeluar

⇒ php artisan make:controller BarangkeluarController

1. Controller dashboard

⇒ php artisan make:controller DashboardController

1. Controller register

⇒ php artisan make:controller RegisterController

1. Controller login

⇒ php artisan make:controller LoginController

1. Mengedit file Controller

Kode program :

<https://github.com/daudfirmann/inventory-projectlaravel/tree/main/app/Http/Controllers>

# Membuat View

1. Membuat view pada direktori /resources/views
2. Membuat view kategori /vkategori
   * + index.blade.php
     + create.blade.php
     + edit.blade.php
     + show.blade.php
3. Membuat view barang pada direktori /vbarang
   * + index.blade.php
     + create.blade.php
     + edit.blade.php
     + show.blade.php
4. Membuat view barangmasuk pada direktori /vbarangmasuk
   * + index.blade.php
     + create.blade.php
     + edit.blade.php
     + show.blade.php
5. Membuat view barangkeluar pada direktori /vbarangkeluar
   * + index.blade.php
     + create.blade.php
     + edit.blade.php
     + show.blade.php
6. Membuat view dashborad
   * + dashboard.blade.php
7. Membuat view register
   * + register.blade.php
8. Membuat view login
   * + login.blade.php
9. Mengedit file view

Kode program :

<https://github.com/daudfirmann/inventory-projectlaravel/tree/main/resources/views>

# Membuat Authentication

* + - 1. Membuat RegisterController

⇒ php artisan make:controller Controller

<?php

namespace *App\Http\Controllers*;

use *Illuminate\Http\Request*;

use *App\Models\User*;

class RegisterController extends *Controller*

{

    public function index(){

        return view('register');

    }

    public function store(*Request* $request){

        $validData = $request->validate([

            'name' => 'required|max:255',

            'email' => 'required|email|unique:users',

            'password' => 'required|min:3|max:255'

        ]);

*User*::create($validData);

        $request->session()->flash('success', 'Registration Successful, Please Login!');

        return redirect('login');

    }

}

* + - 1. Membuat LoginController

⇒ php artisan make:controller LoginController

<?php

namespace *App\Http\Controllers*;

use *Illuminate\Http\Request*;

use *Illuminate\Support\Facades\Auth*;

class LoginController extends *Controller*

{

    public function index() {

        return view('login');

    }

    public function authenticate(*Request* $request) {

        $credentials = $request->validate([

            'email' => 'required|email',

            'password' => 'required'

        ]);

        if (*Auth*::attempt($credentials)) {

            $request->session()->regenerate();

            return redirect()->intended('dashboard');

        }

        return back()->with('loginError', 'Login failed!');

    }

    public function logout(){

*Auth*::logout();

        request()->session()->invalidate();

        request()->session()->regenerateToken();

        return redirect('login');

    }

}

* + - 1. Membuat view register

⇒ mkdir /var/www/jagokoding/resources/register.blade.php

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="utf-8">

    <meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1, shrink-to-fit=no">

    <meta *name*="description" *content*="">

    <meta *name*="author" *content*="">

    <title>JagoKoding</title>

    <!-- Custom fonts -->

    <link *href*="{{ asset('theme/vendor/fontawesome-free/css/all.min.css') }}" *rel*="stylesheet" *type*="text/css">

    <link *href*="https://fonts.googleapis.com/css?family=Nunito:200,200i,300,300i,400,400i,600,600i,700,700i,800,800i,900,900i" *rel*="stylesheet">

    <!-- Custom styles -->

    <link *href*="{{ asset('theme/css/sb-admin-2.min.css') }}" *rel*="stylesheet">

</head>

<body *class*="bg-gradient-primary">

    <div *class*="container">

        <br>

        <br>

        <!-- Outer Row -->

        <div *class*="row justify-content-center">

            <div *class*="col-xl-10 col-lg-12 col-md-9">

                <div *class*="card o-hidden border-0 shadow-lg my-5">

                    <div *class*="card-body p-0">

                        <!-- Nested Row within Card Body -->

                        <div *class*="row">

                            <div *class*="col-lg-6 d-none d-lg-block bg-register-image"></div>

                            <div *class*="col-lg-6">

                                <div *class*="p-5">

                                    <div *class*="text-center">

                                        <h1 *class*="h4 text-gray-900 mb-4">Welcome!</h1>

                                    </div>

                                    <form *class*="user" *action*="register" *method*="POST">

                                        @csrf

                                        <div *class*="form-group">

                                            <input *type*="text" *class*="form-control form-control-user @error('name') is-invalid @enderror" *name*="name" *placeholder*="Name" *required* *value*="{{ old('name') }}">

                                            @error('name')

                                                <div *class*="invalid-feedback">{{ $message }}</div>

                                            @enderror

                                        </div>

                                        <div *class*="form-group">

                                            <input *type*="email" *class*="form-control form-control-user @error('email') is-invalid @enderror" *name*="email" *placeholder*="Email Address" *required* *value*="{{ old('email') }}">

                                            @error('email')

                                                <div *class*="invalid-feedback">{{ $message }}</div>

                                            @enderror

                                        </div>

                                        <div *class*="form-group">

                                            <input *type*="password" *class*="form-control form-control-user @error('password') is-invalid @enderror" *name*="password" *placeholder*="Password" *required*>

                                            @error('password')

                                                <div *class*="invalid-feedback">{{ $message }}</div>

                                            @enderror

                                        </div>

                                        <div *class*="form text-center">

                                            <button *type*="submit" *class*="btn btn-primary">Register</button>

                                        </div>

                                    </form>

                                    <hr>

                                    <div *class*="text-center">

                                        <a *class*="small" *href*="/login">Have an Account? Login Now!</a>

                                    </div>

                                </div>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <script *src*="vendor/jquery/jquery.min.js"></script>

    <script *src*="vendor/bootstrap/js/bootstrap.bundle.min.js"></script>

    <script *src*="vendor/jquery-easing/jquery.easing.min.js"></script>

    <script *src*="js/sb-admin-2.min.js"></script>

</body>

</html>

* + - 1. Membuat view login

⇒ mkdir /var/www/jagokoding/resources/login.blade.php

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="utf-8">

    <meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1, shrink-to-fit=no">

    <meta *name*="description" *content*="">

    <meta *name*="author" *content*="">

    <title>JagoKoding</title>

    <!-- Custom fonts -->

    <link *href*="{{ asset('theme/vendor/fontawesome-free/css/all.min.css') }}" *rel*="stylesheet" *type*="text/css">

    <link *href*="https://fonts.googleapis.com/css?family=Nunito:200,200i,300,300i,400,400i,600,600i,700,700i,800,800i,900,900i" *rel*="stylesheet">

    <!-- Custom styles -->

    <link *href*="{{ asset('theme/css/sb-admin-2.min.css') }}" *rel*="stylesheet">

</head>

<body *class*="bg-gradient-primary">

    <div *class*="container">

        <br>

        <br>

        <!-- Outer Row -->

        <div *class*="row justify-content-center">

            <div *class*="col-xl-10 col-lg-12 col-md-9">

                <div *class*="card o-hidden border-0 shadow-lg my-5">

                    <div *class*="card-body p-0">

                        <!-- Nested Row within Card Body -->

                        <div *class*="row">

                            <div *class*="col-lg-6 d-none d-lg-block bg-login-image"></div>

                            <div *class*="col-lg-6">

                                <div *class*="p-5">

                                    @if(session()->has('success'))

                                    <div *class*="alert alert-success" *role*="alert">

                                        {{ session('success') }}

                                    </div>

                                    @endif

                                    @if(session()->has('loginError'))

                                    <div *class*="alert alert-danger" *role*="alert">

                                        {{ session('loginError') }}

                                    </div>

                                    @endif

                                    <div *class*="text-center">

                                        <h1 *class*="h4 text-gray-900 mb-4">Welcome Back!</h1>

                                    </div>

                                    <form *class*="user" *action*="login" *method*="POST">

                                        @csrf

                                        <div *class*="form-group">

                                            <input *type*="email" *class*="form-control form-control-user @error('email') is-invalid @enderror" *name*="email" *placeholder*="Email Address" *autofocus* *required* *value*="{{ old('email') }}">

                                            @error('email')

                                                <div *class*="invalid-feedback">{{ $message }}</div>

                                            @enderror

                                        </div>

                                        <div *class*="form-group">

                                            <input *type*="password" *class*="form-control form-control-user" *name*="password" *placeholder*="Password" *required*>

                                        </div>

                                        <div *class*="form text-center">

                                            <button *type*="submit" *class*="btn btn-primary">Login</button>

                                        </div>

                                    </form>

                                    <hr>

                                    <div *class*="text-center">

                                        <a *class*="small" *href*="/register">Not Registered? Create an Account!</a>

                                    </div>

                                </div>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <script *src*="vendor/jquery/jquery.min.js"></script>

    <script *src*="vendor/bootstrap/js/bootstrap.bundle.min.js"></script>

    <script *src*="vendor/jquery-easing/jquery.easing.min.js"></script>

    <script *src*="js/sb-admin-2.min.js"></script>

</body>

</html>

# Membuat Route

Membuat route pada direktori /routes/web.php

<?php

use *Illuminate\Support\Facades\Route*;

use *App\Http\Controllers\BarangController*;

use *App\Http\Controllers\KategoriController*;

use *App\Http\Controllers\BarangmasukController*;

use *App\Http\Controllers\BarangkeluarController*;

use *App\Http\Controllers\LoginController*;

use *App\Http\Controllers\RegisterController*;

use *App\Http\Controllers\DashboardController*;

/\*

|--------------------------------------------------------------------------

| Web Routes

|--------------------------------------------------------------------------

|

| Here is where you can register web routes for your application. These

| routes are loaded by the RouteServiceProvider and all of them will

| be assigned to the "web" middleware group. Make something great!

|

\*/

*Route*::get('/', function () {

    return view('dashboard');

});

*Route*::get('dashboard', [*DashboardController*::class, 'index'])->name('dashboard');

*Route*::resource('barang', *BarangController*::class)->middleware('auth');

*Route*::resource('kategori', *KategoriController*::class)->middleware('auth');

*Route*::resource('barangmasuk', *BarangmasukController*::class)->middleware('auth');

*Route*::resource('barangkeluar', *BarangkeluarController*::class)->middleware('auth');

*Route*::get('login', [*LoginController*::class, 'index'])->name('login')->middleware('guest');

*Route*::post('login', [*LoginController*::class, 'authenticate']);

*Route*::post('logout', [*LoginController*::class, 'logout'])->name('logout');

*Route*::get('register', [*RegisterController*::class, 'index'])->middleware('guest');

*Route*::post('register', [*RegisterController*::class, 'store']);

# Membuat Store Function

Membuat function ketKategori() untuk menampilkan detail field kategori dari tabel kategori

DELIMITER $$

CREATE FUNCTION ketKategori(kat VARCHAR(4))

RETURNS varchar(30)

BEGIN

DECLARE kategori VARCHAR(30);

IF kat = 'M' THEN

SET kategori = "Modal";

ELSEIF kat="A" THEN

SET kategori = "Alat";

ELSEIF kat="BHP" THEN

SET kategori = "Bahan Habis Pakai";

ELSEIF kat="BTHP" THEN

SET kategori = "Bahan Tidak Habis Pakai";

END IF;

RETURN kategori;

END$$

DELIMITER ;

# Membuat Store Procedure

Membuat procedure getKategoriAll() untuk mengakses seluruh record dari tabel kategori

DELIMITER $$

CREATE PROCEDURE getKategoriAll()

BEGIN

SELECT \* FROM kategori;

END$$

DELIMITER ;

# Membuat Trigger

1. Membuat trigger barang\_stokplus untuk menambah stok di tabel barang dengan cara menambah record di tabel barangmasuk

DELIMITER $$

CREATE TRIGGER barang\_stokplus

AFTER INSERT ON barangmasuk

FOR EACH ROW

BEGIN

UPDATE barang SET barang.stok = barang.stok + NEW.qty\_masuk WHERE barang.id=NEW.barang\_id;

END;

$$

DELIMITER ;

1. Membuat trigger update\_barangmasuk untuk mengedit stok di tabel barang dengan cara mengedit record di tabel barangmasuk

DELIMITER //

CREATE TRIGGER update\_barangmasuk

AFTER UPDATE ON barangmasuk

FOR EACH ROW

BEGIN

DECLARE qty\_diff INT;

SET qty\_diff = NEW.qty\_masuk - OLD.qty\_masuk;

SET stok = stok + qty\_diff

WHERE id = NEW.barang\_id;

END;

//

DELIMITER ;

1. Membuat trigger delete\_barangmasuk untuk menghapus stok di tabel barang dengan cara menghapus record di tabel barangmasuk

DELIMITER //

CREATE TRIGGER delete\_barangmasuk

AFTER DELETE ON barangmasuk

FOR EACH ROW

BEGIN

UPDATE barang

SET stok = stok - OLD.qty\_masuk

WHERE id = OLD.barang\_id;

END;

//

DELIMITER ;

1. Mambuat trigger barang\_stokmin untuk mengurangi stok di tabel barang dengan cara menambah record di tabel barangkeluar

DELIMITER $$

CREATE TRIGGER barang\_stokmin

AFTER INSERT ON barangkeluar

FOR EACH ROW

BEGIN

UPDATE barang SET barang.stok = barang.stok - NEW.qty\_keluar WHERE barang.id=NEW.barang\_id;

END;

$$

DELIMITER ;

1. Membuat trigger update\_barangkeluar untuk mengedit stok di tabel barang dengan cara mengedit record di tabel barangkeluar

DELIMITER //

CREATE TRIGGER update\_barangkeluar

AFTER UPDATE ON barangkeluar

FOR EACH ROW

BEGIN

DECLARE qty\_diff INT;

SET qty\_diff = NEW.qty\_keluar - OLD.qty\_keluar;

UPDATE barang

SET stok = stok - qty\_diff

WHERE id = NEW.barang\_id;

END;

//

DELIMITER ;

1. Membuat trigger delete\_barangkeluar untuk menghapus stok di tabel barang dengan cara menghapus record di tabel barangkeluar

DELIMITER //

CREATE TRIGGER delete\_barangkeluar

AFTER DELETE ON barangkeluar

FOR EACH ROW

BEGIN

UPDATE barang

SET stok = stok + OLD.qty\_keluar

WHERE id = OLD.barang\_id;

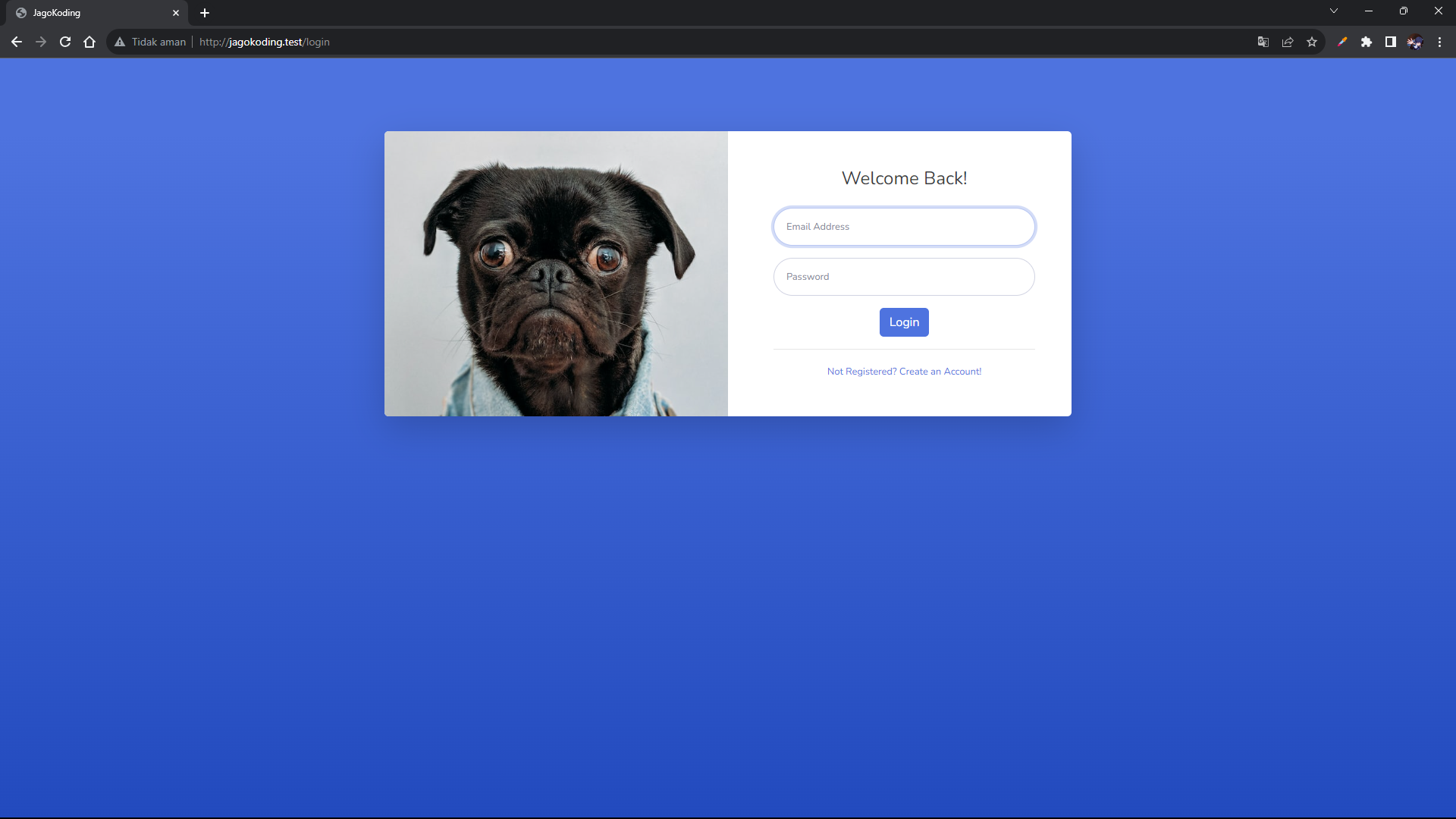
END;

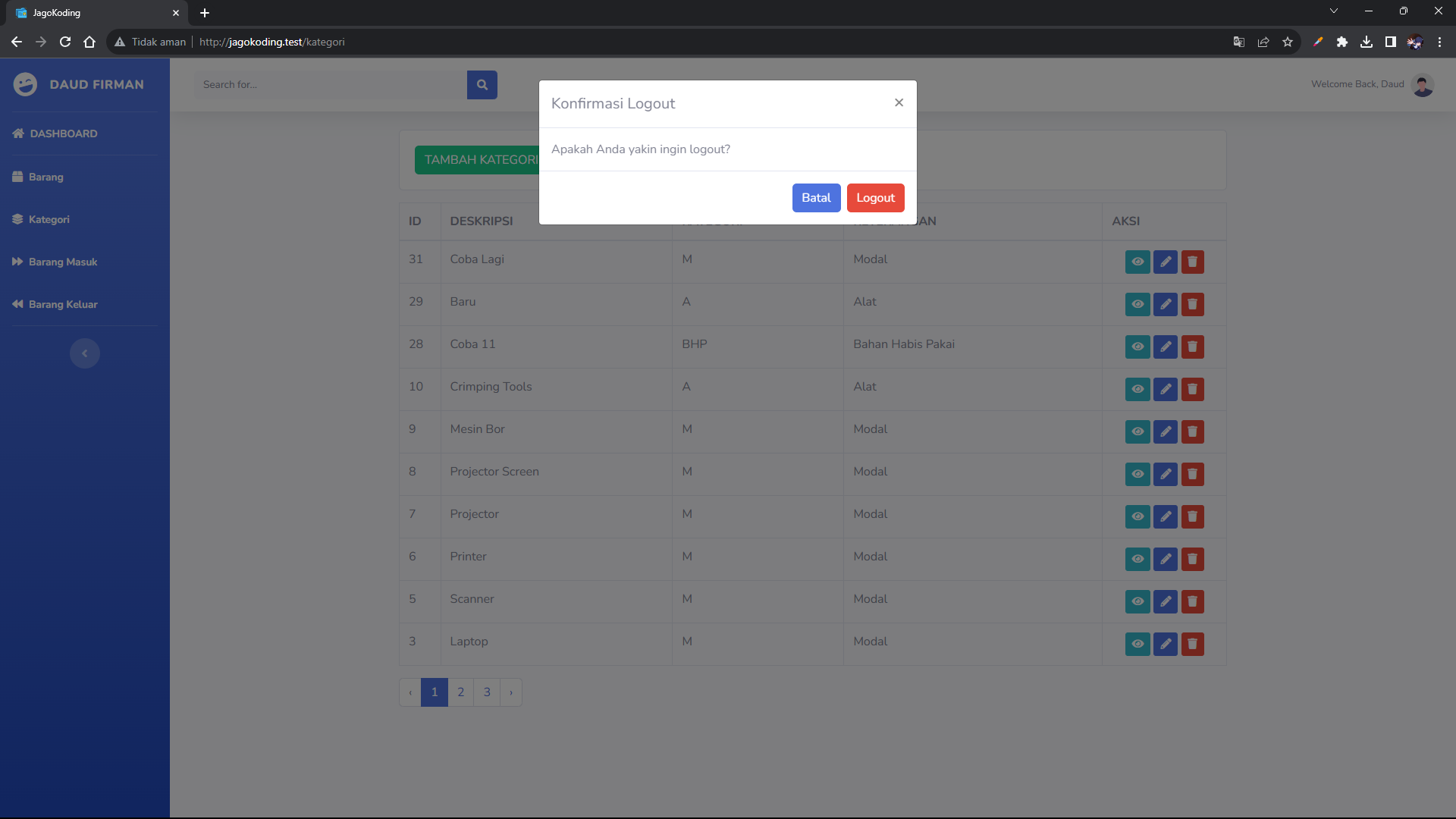
//

DELIMITER ;

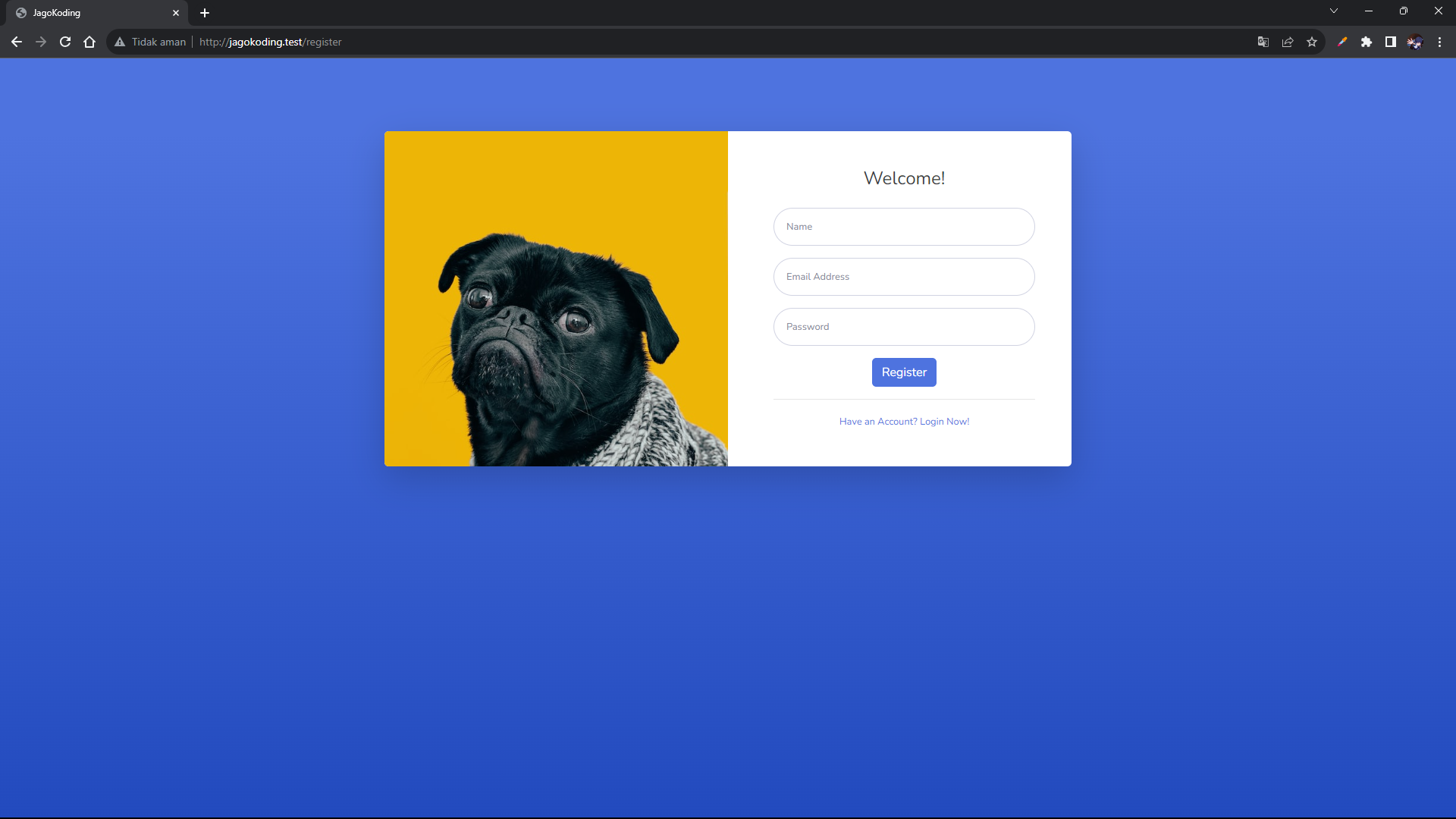
# HASIL

* Halaman Login/Logout

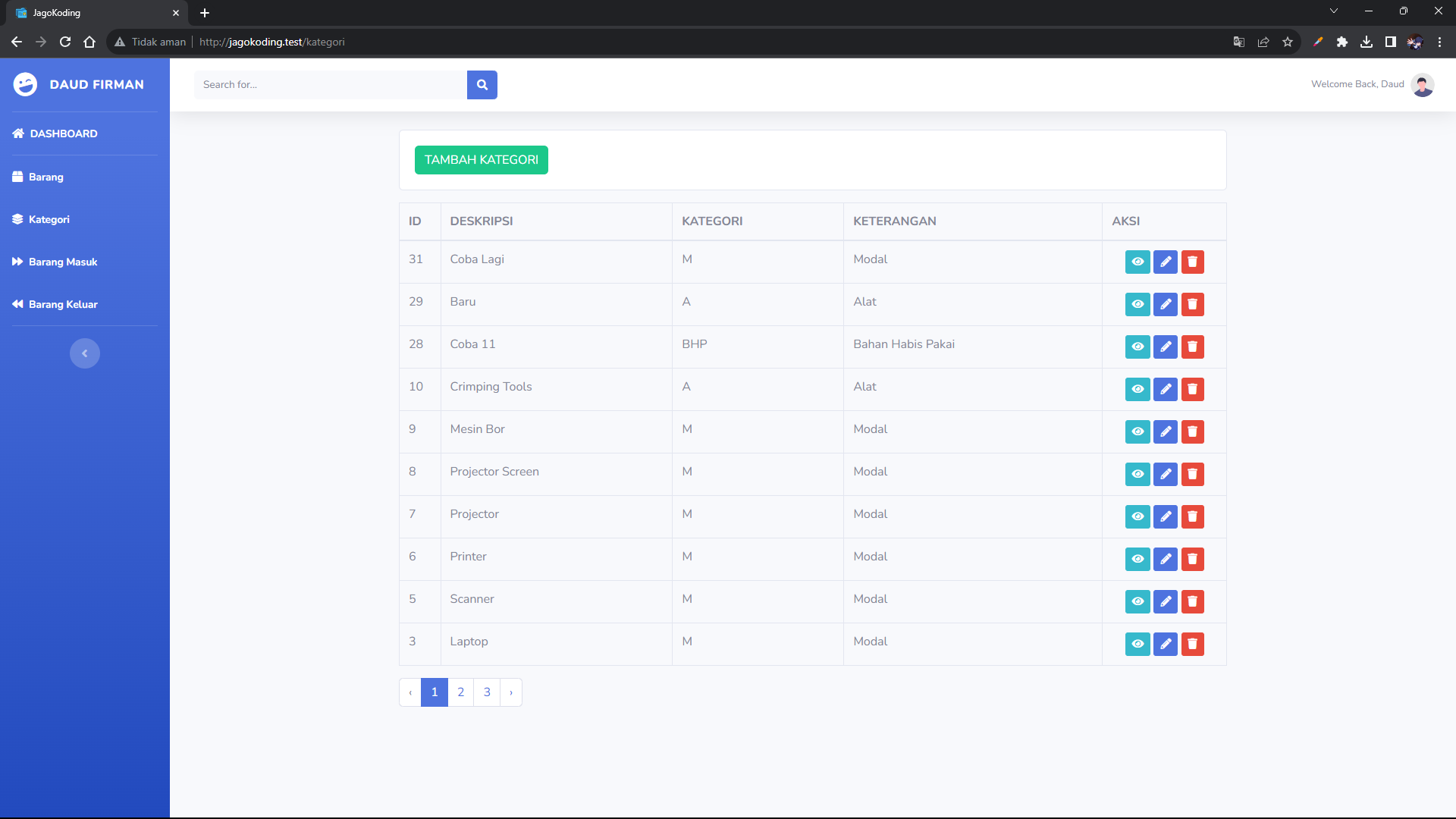




* Halaman Registrasi



* Halaman Kategori

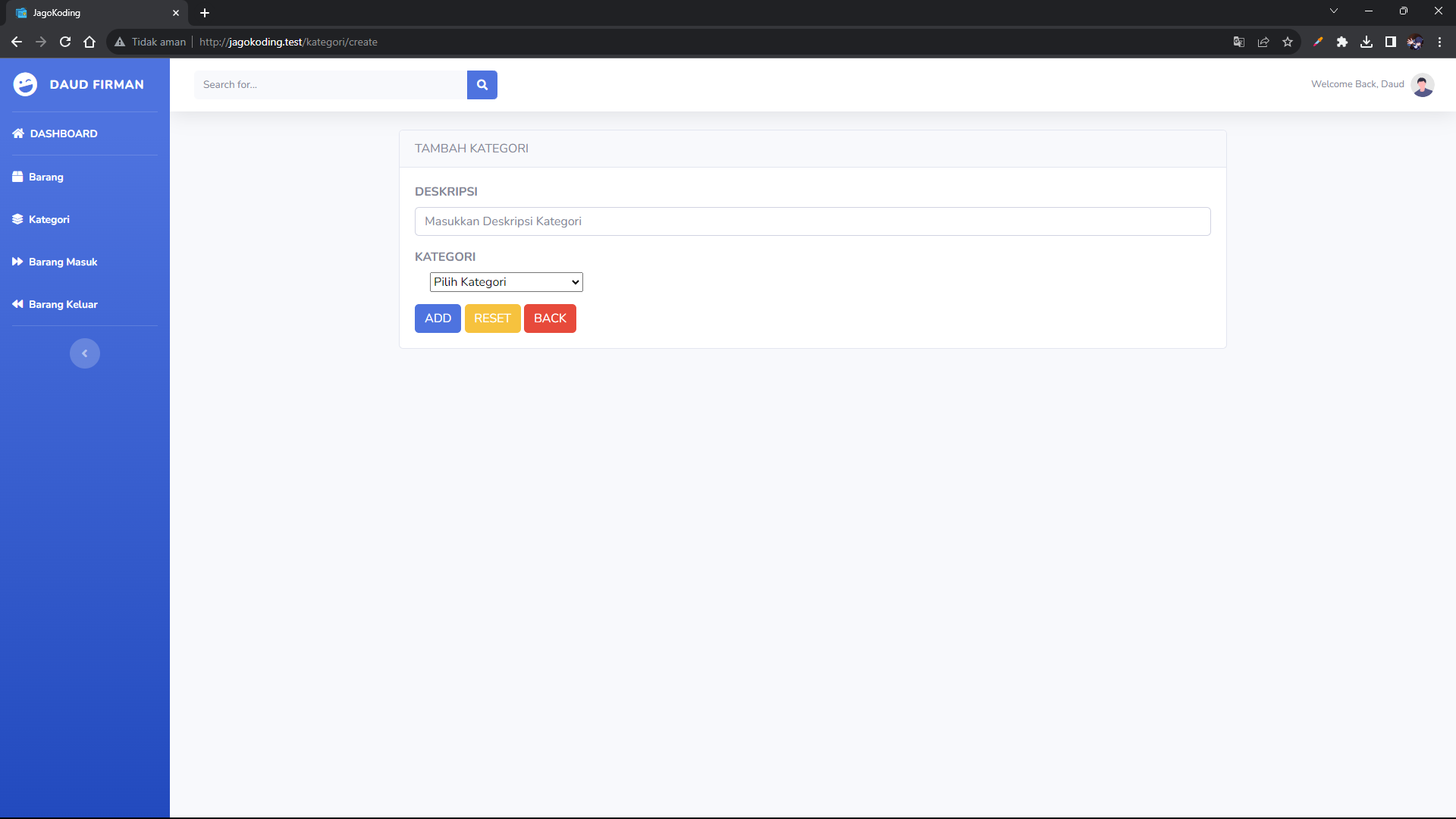


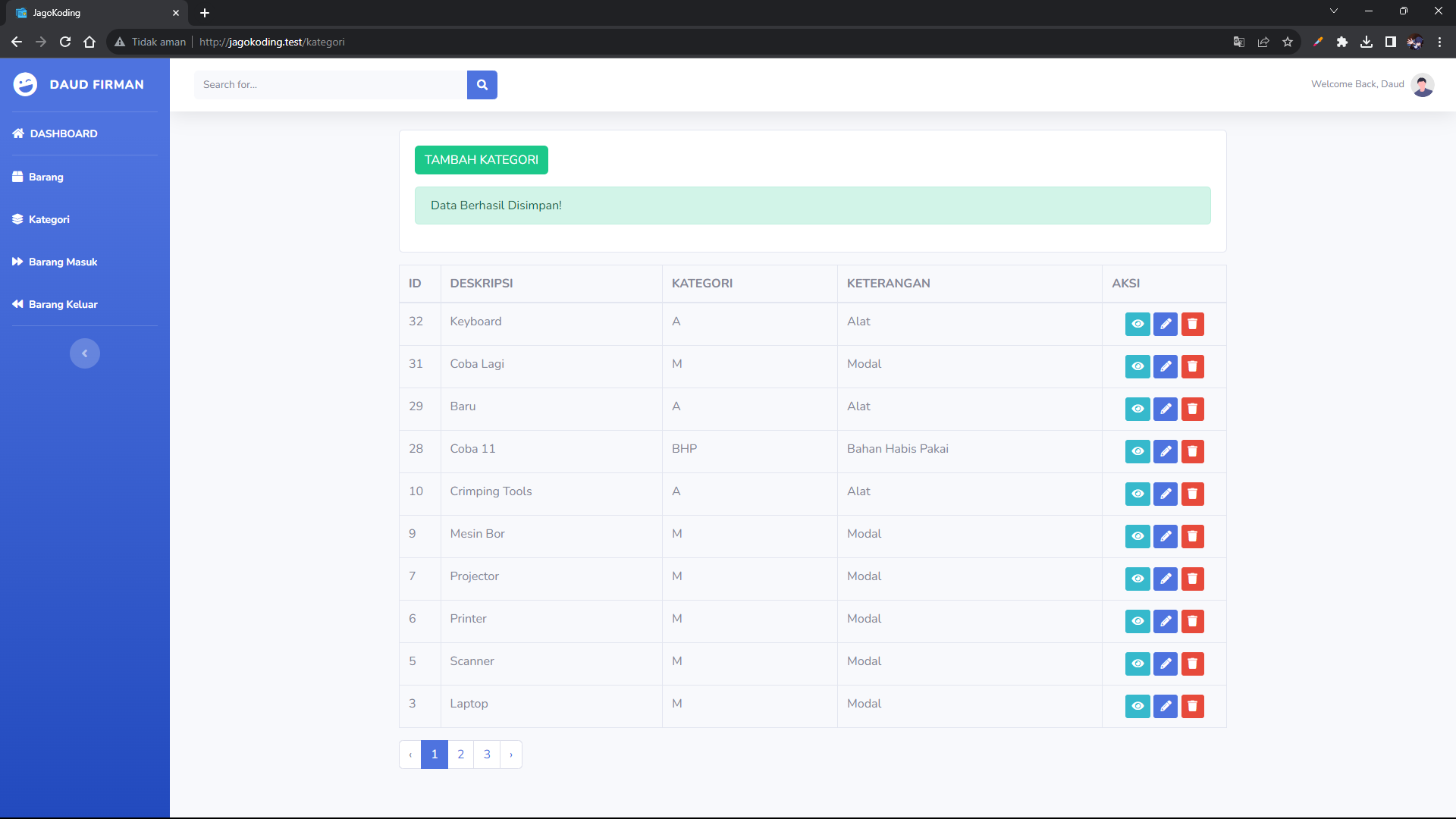
* + Show



* + New

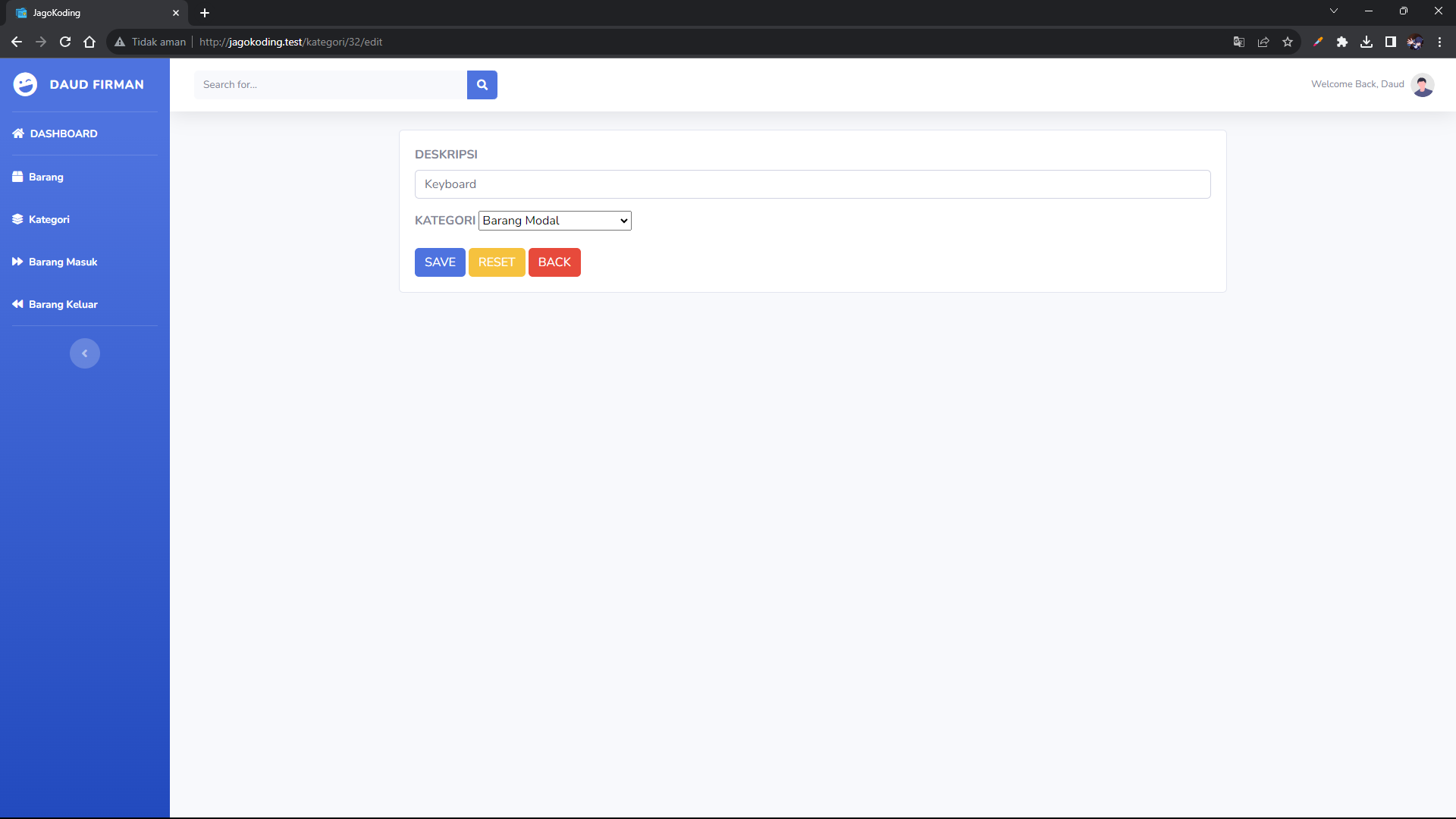
⇒ Menambah kategori baru, keyboard

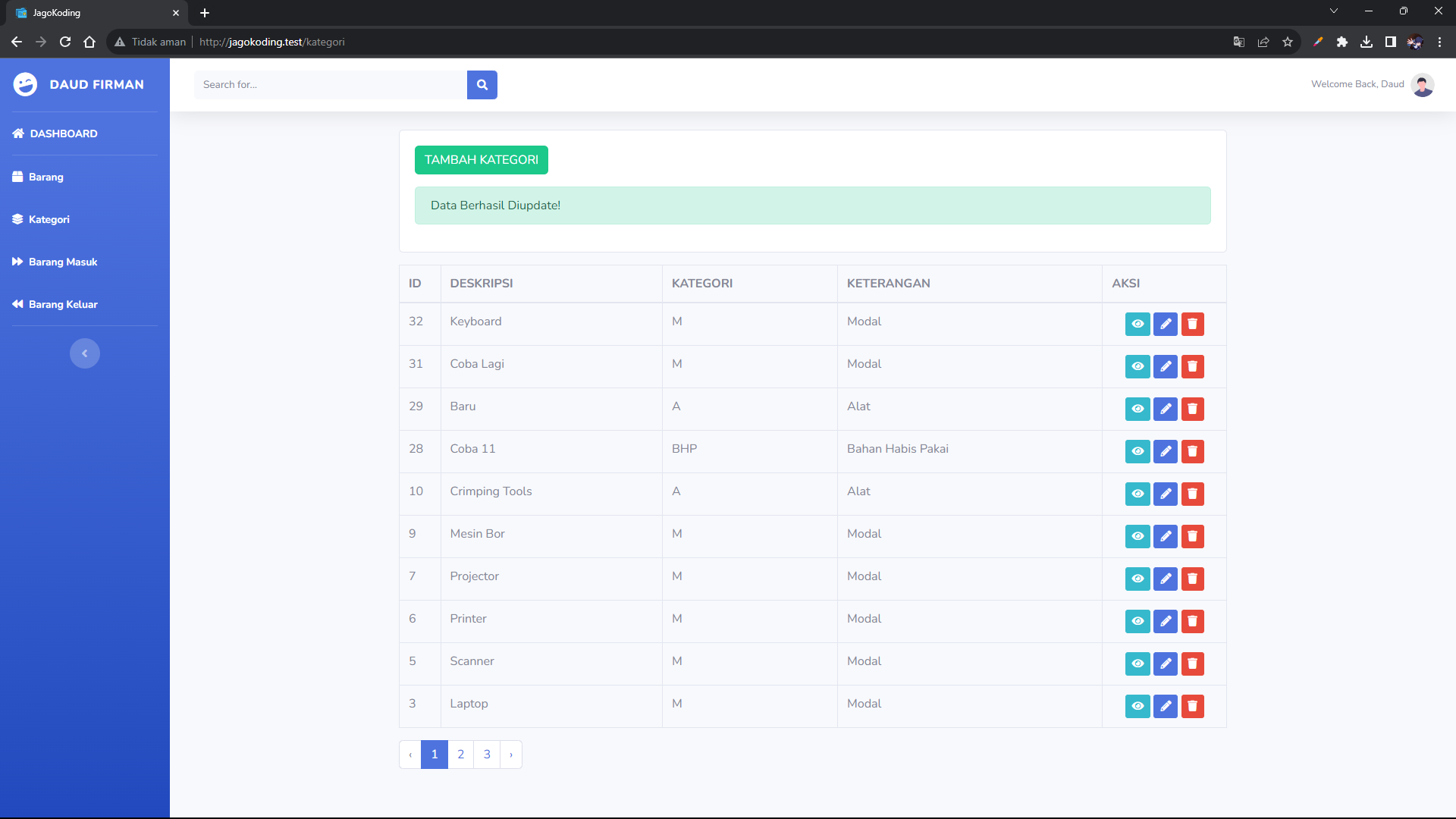




* + Edit

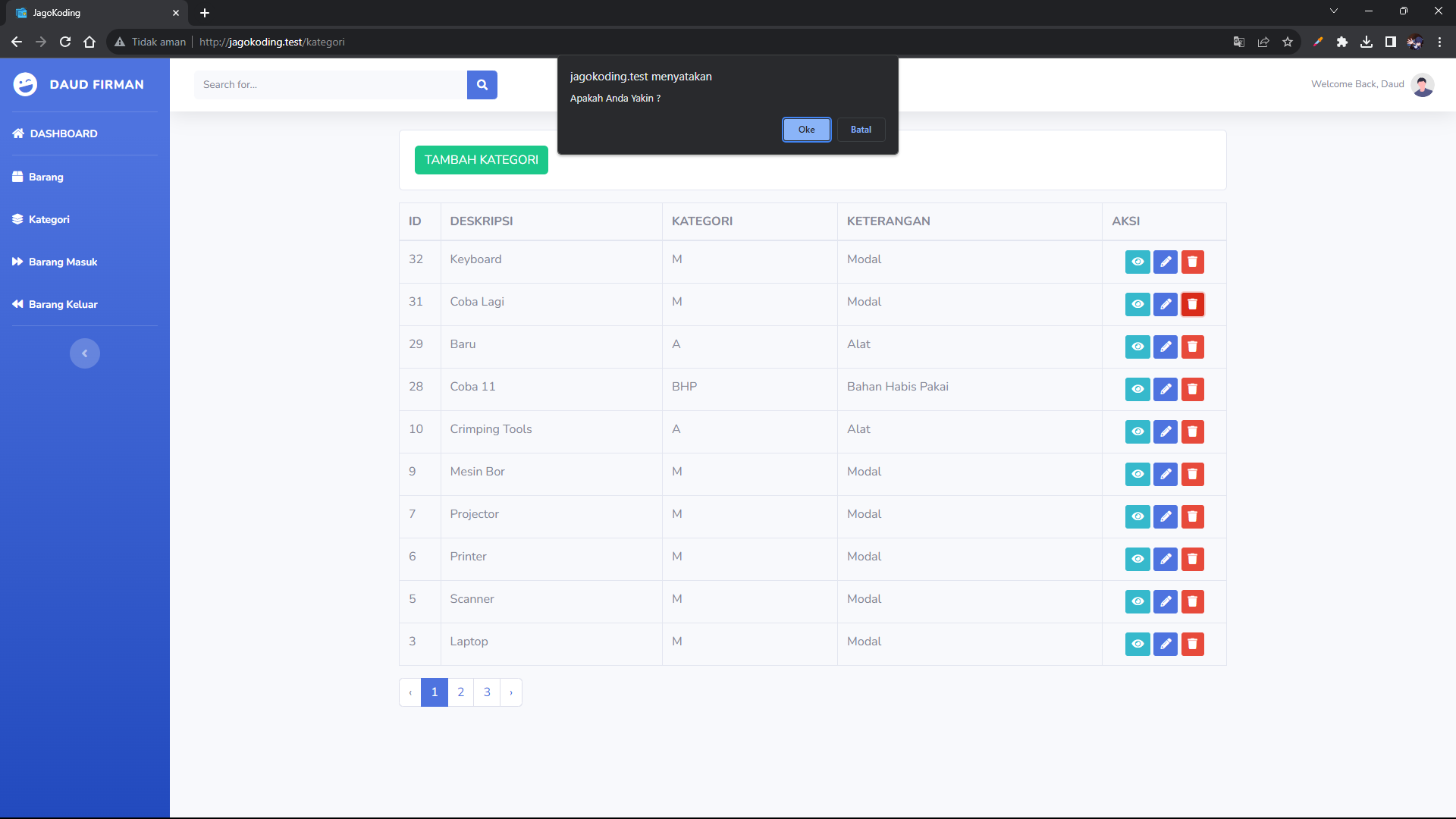
⇒ Mengedit kategori keyboard, dari Alat menjadi Modal

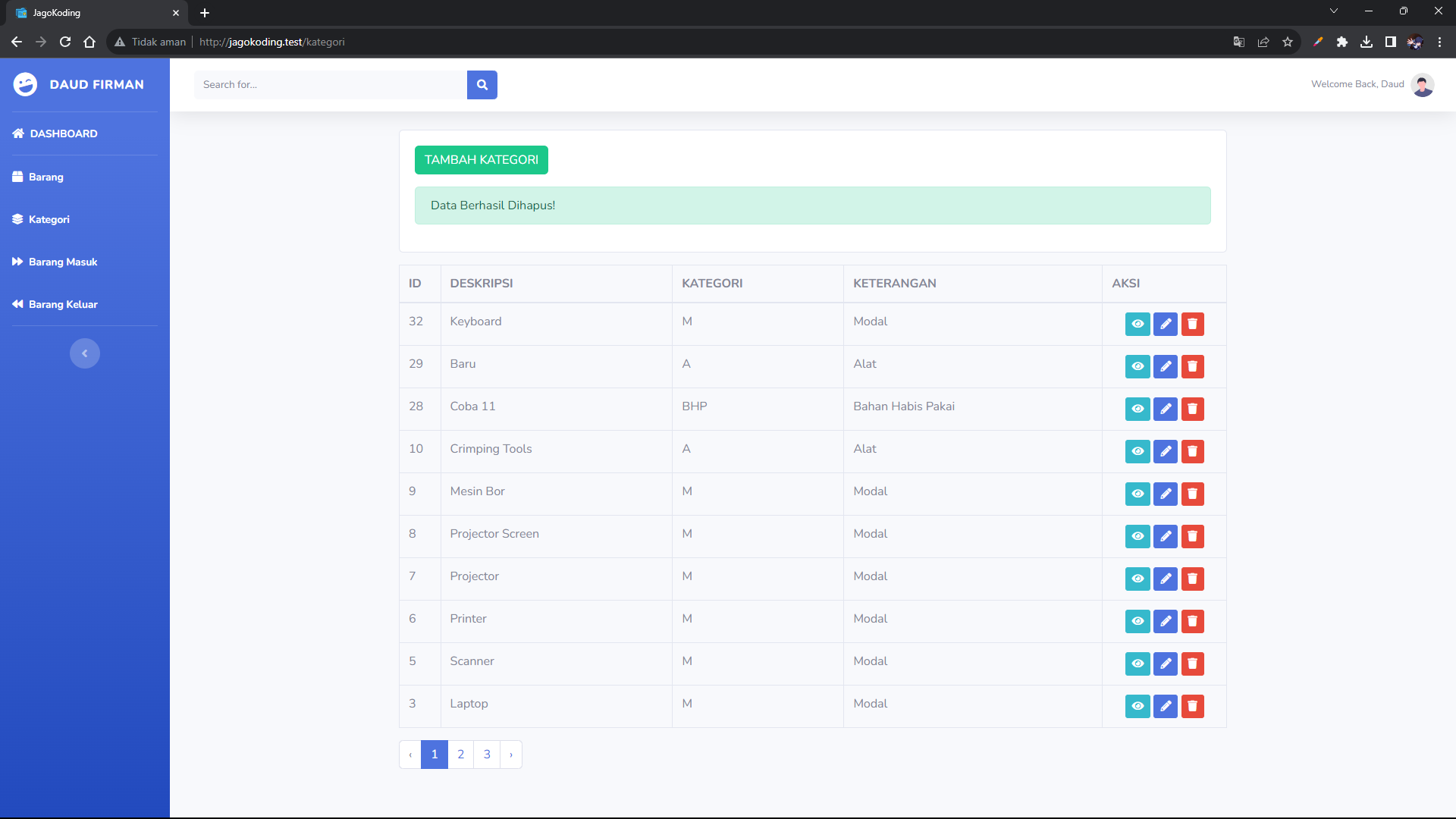




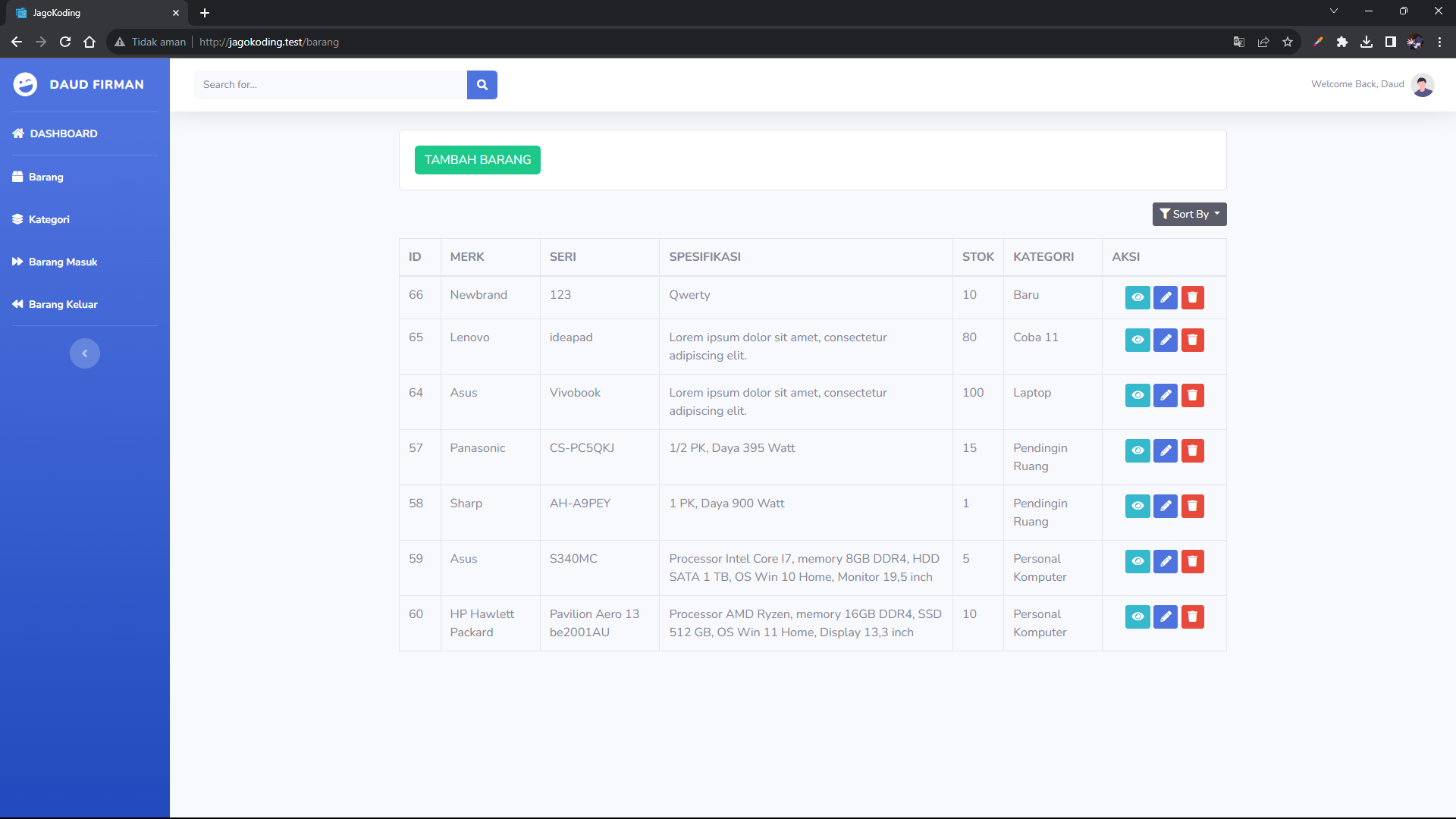
* + Delete

⇒ Menghapus kategori





* Halaman Barang

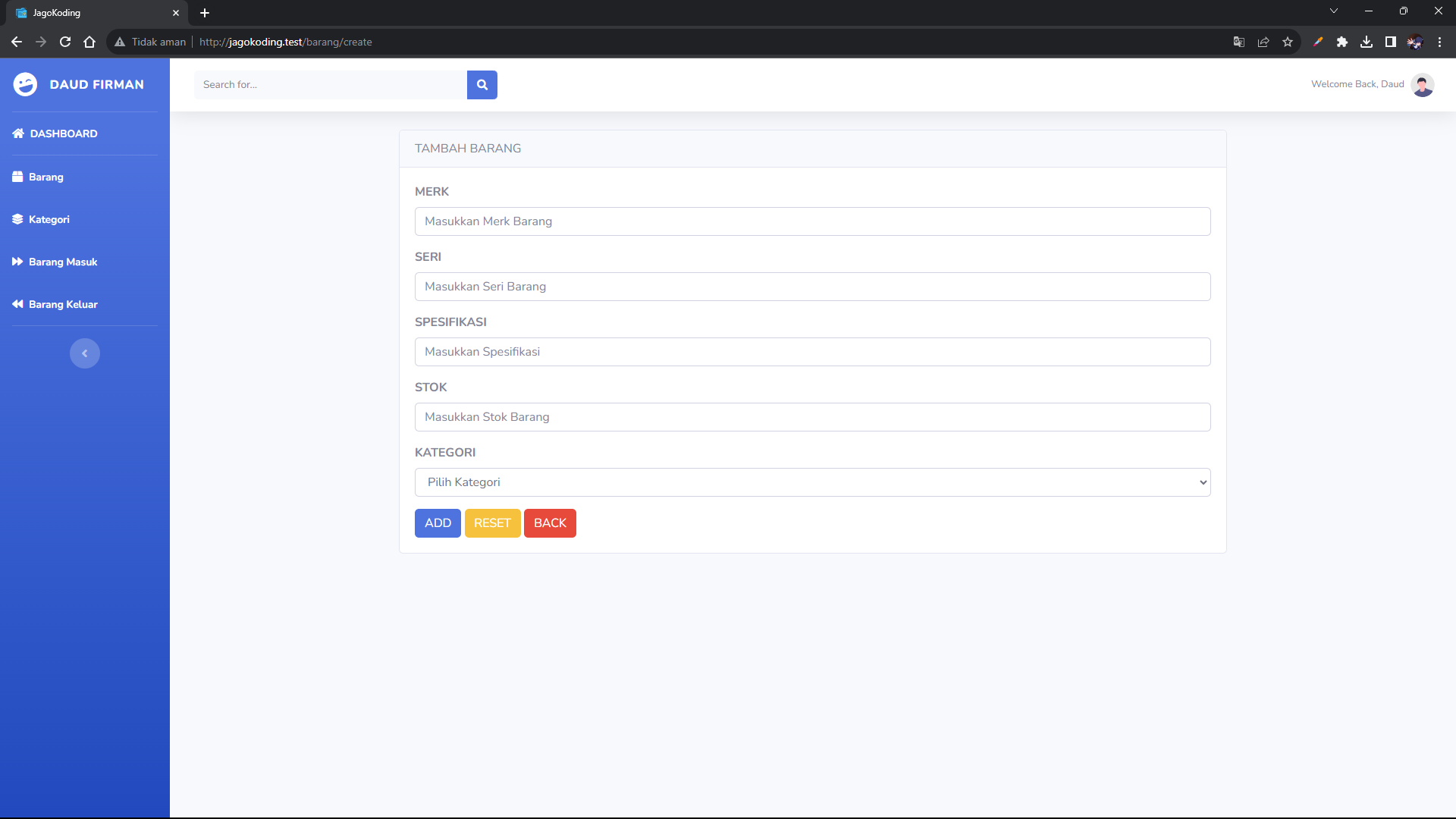


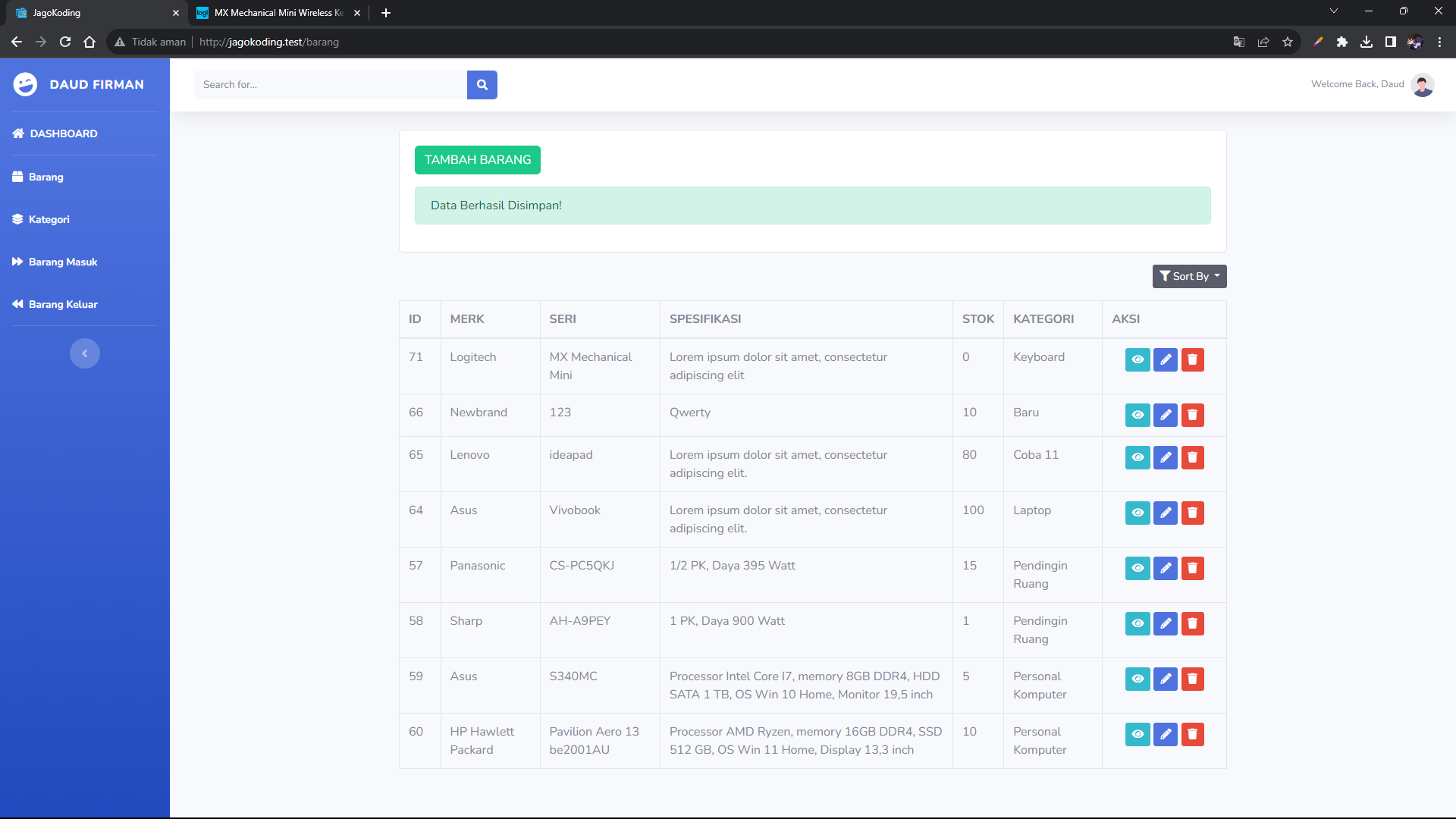
* + Show



* + New

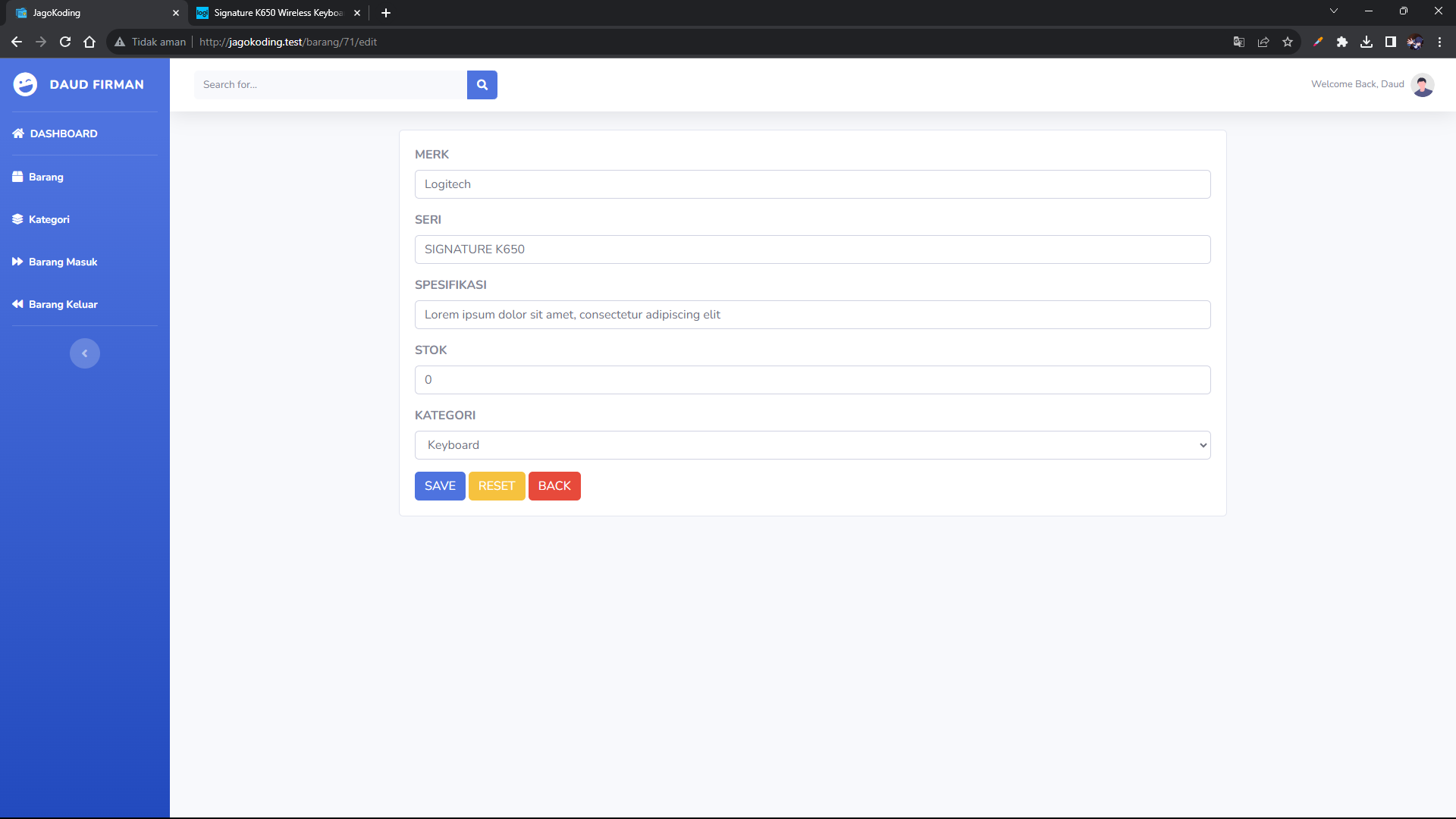
⇒ Menambah barang baru, merk Logitech, kategori keyboard

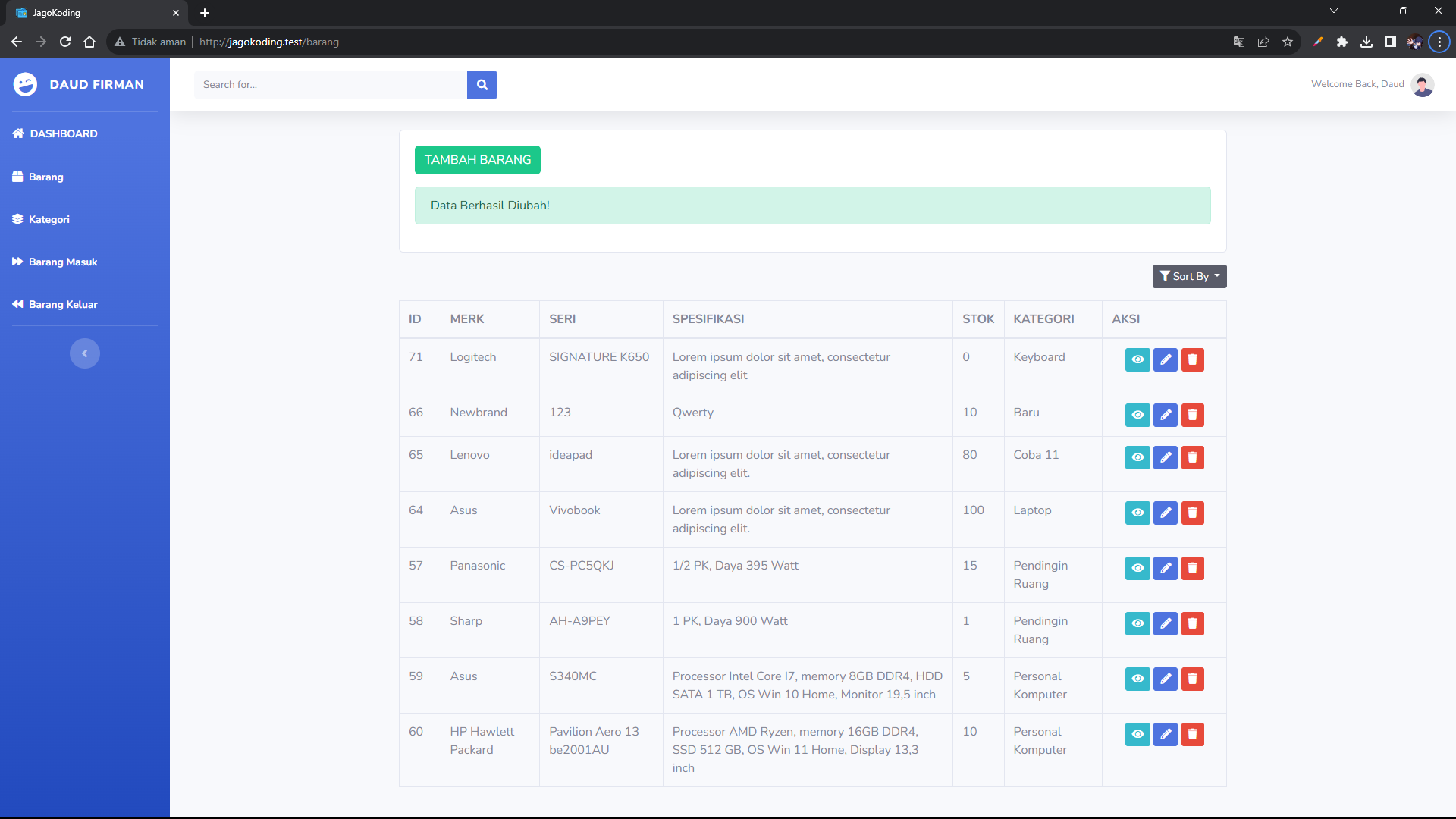




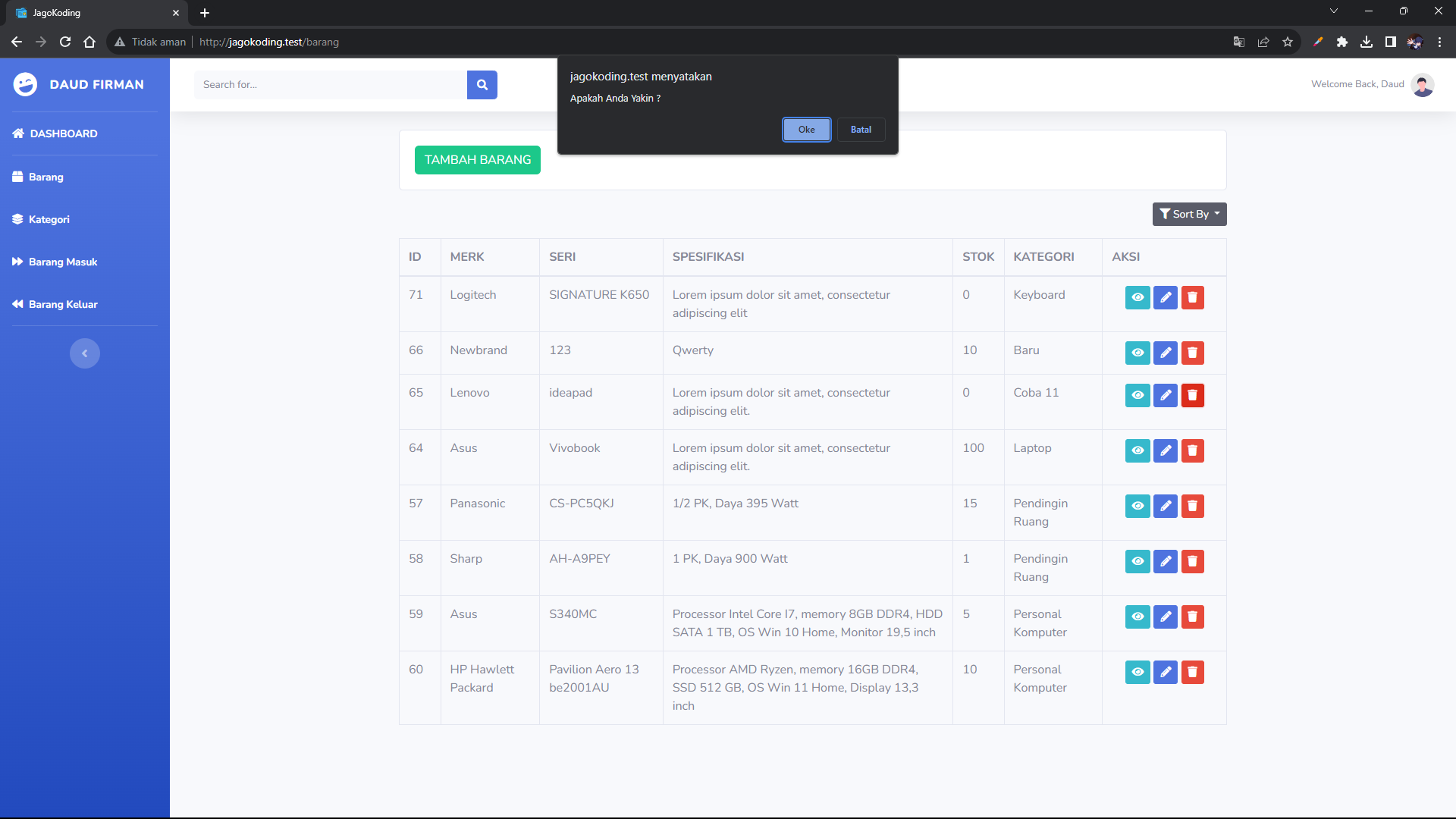
* + Edit

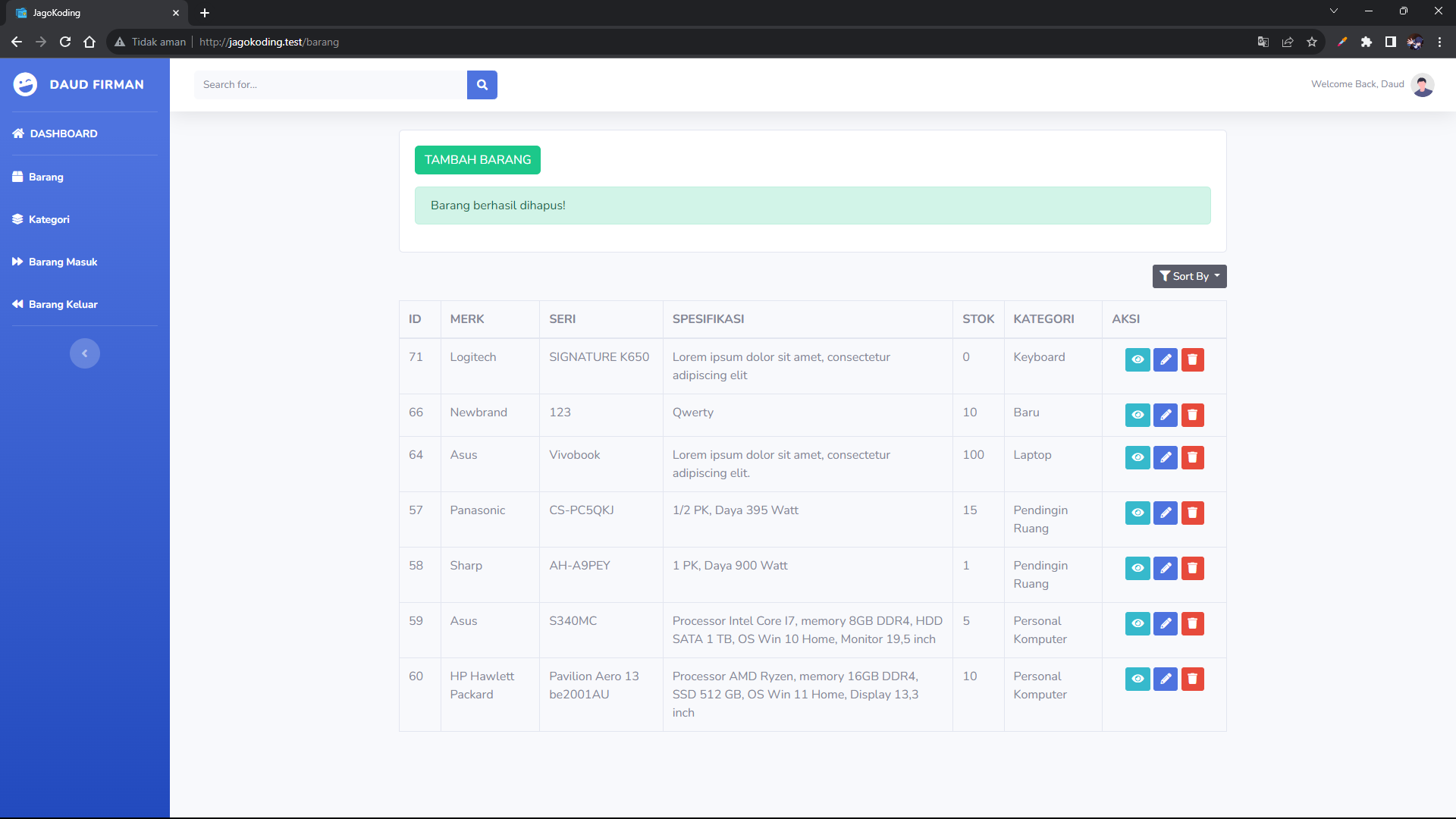
⇒ Mengedit seri



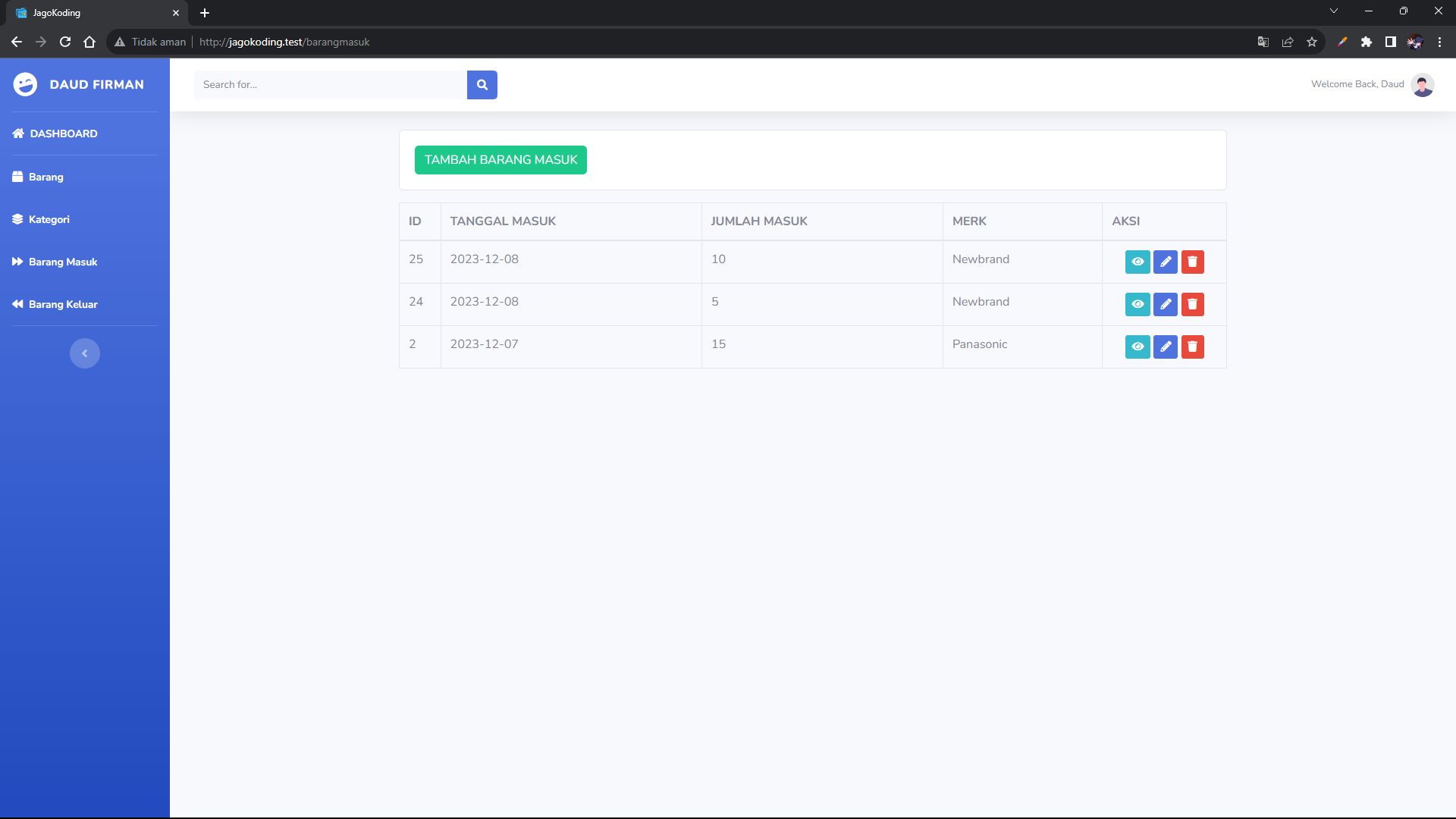


* + Delete

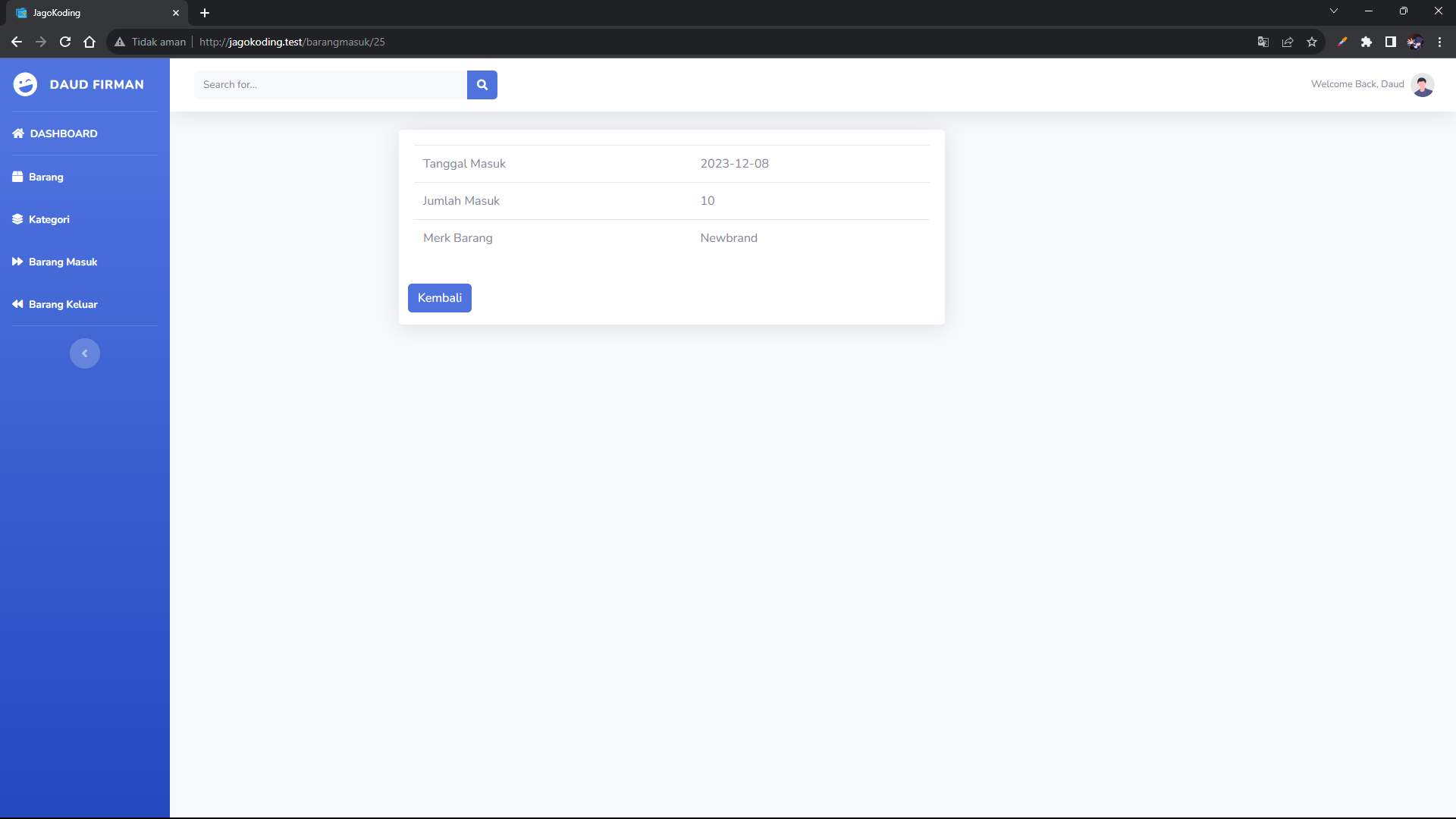




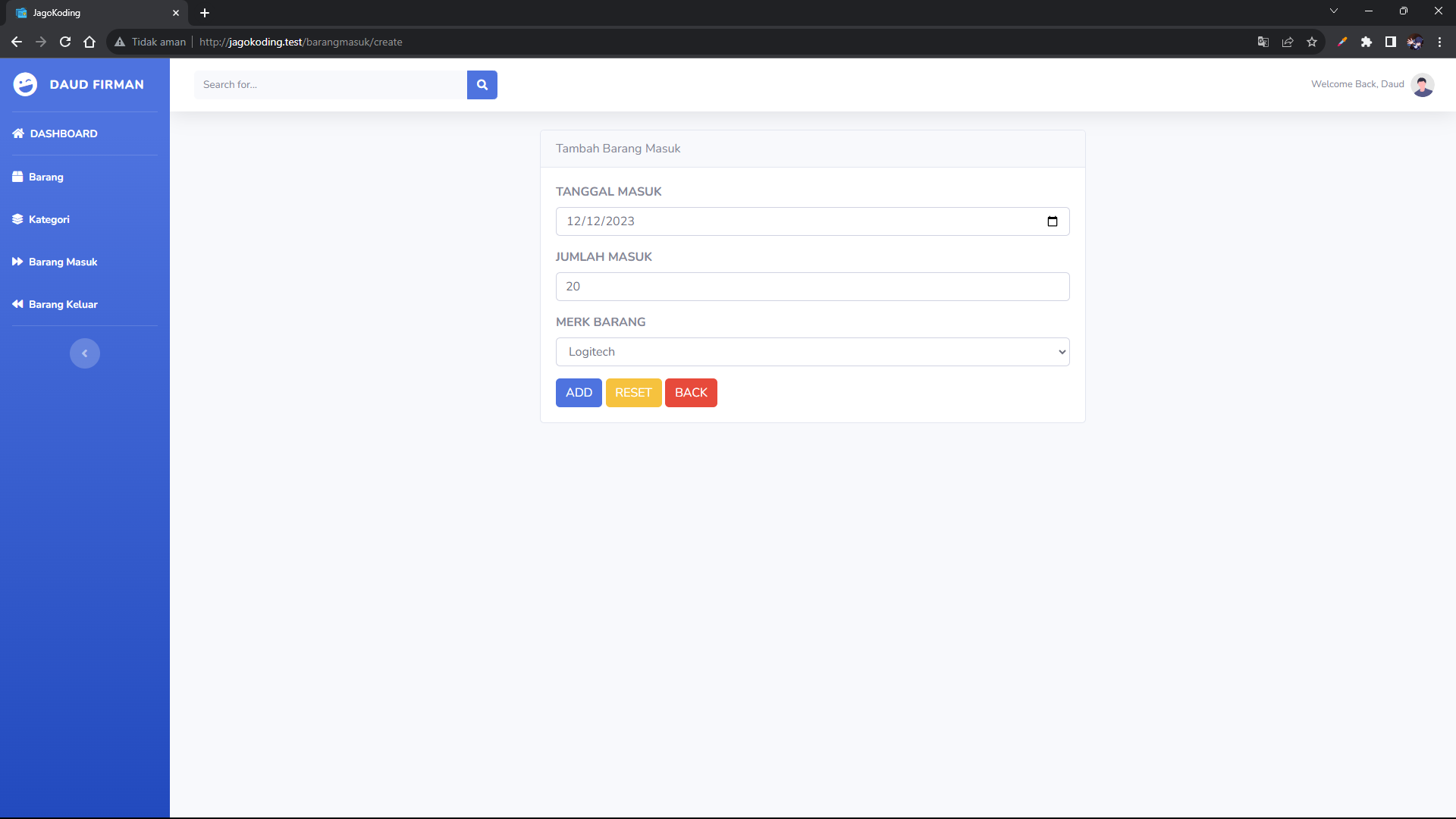
* Halaman Barang Masuk

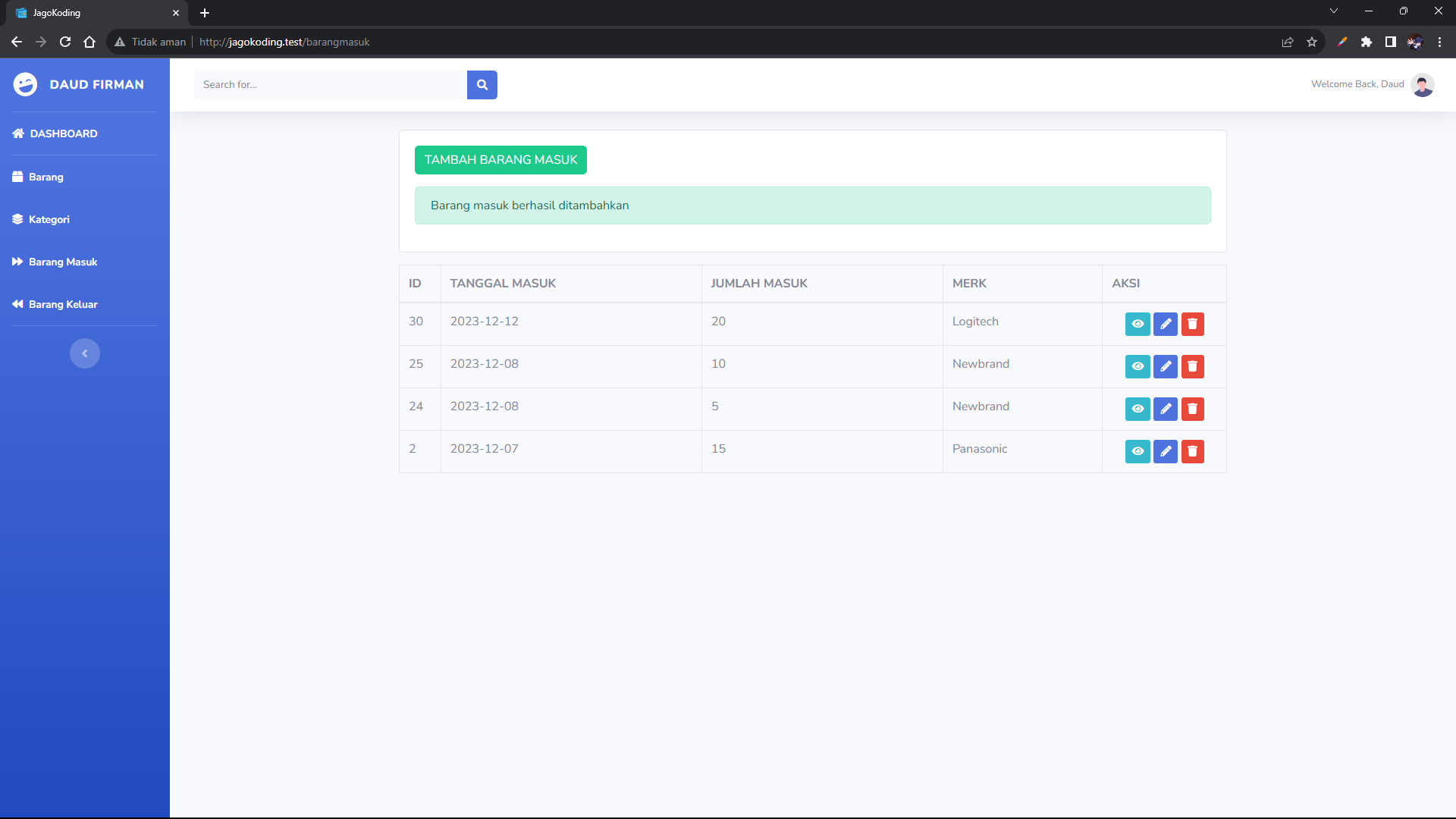


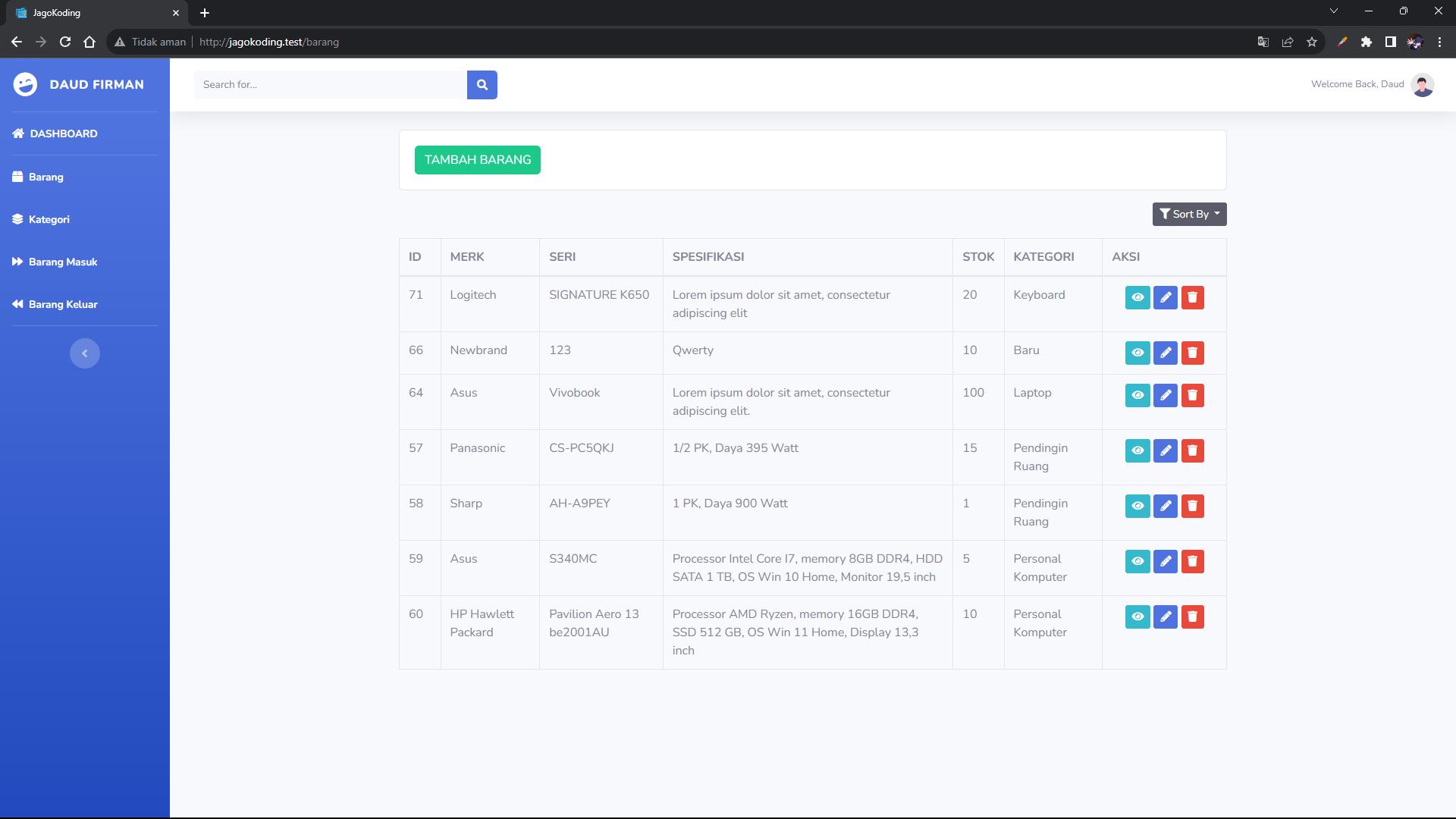
* + Show



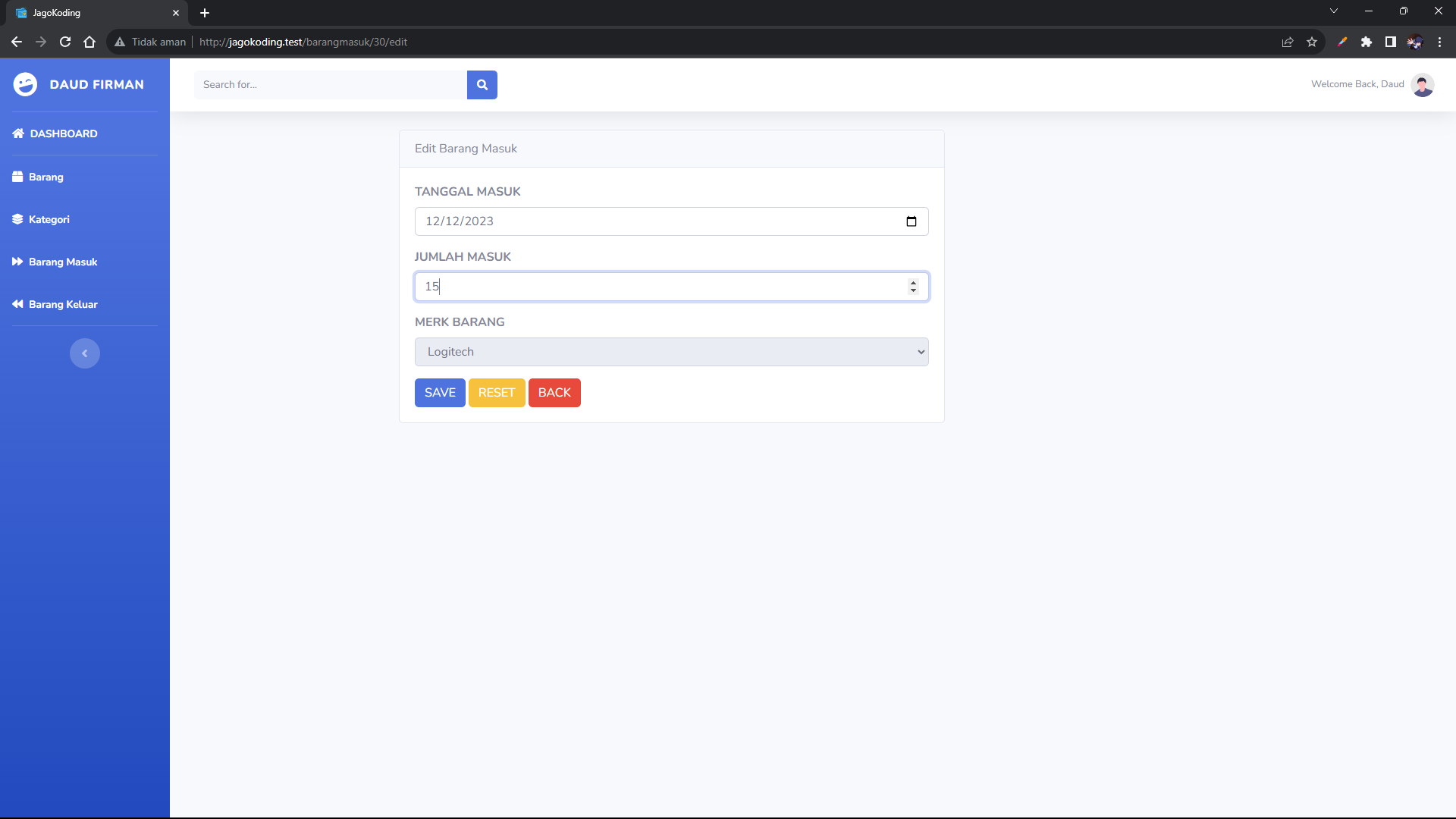
* + New

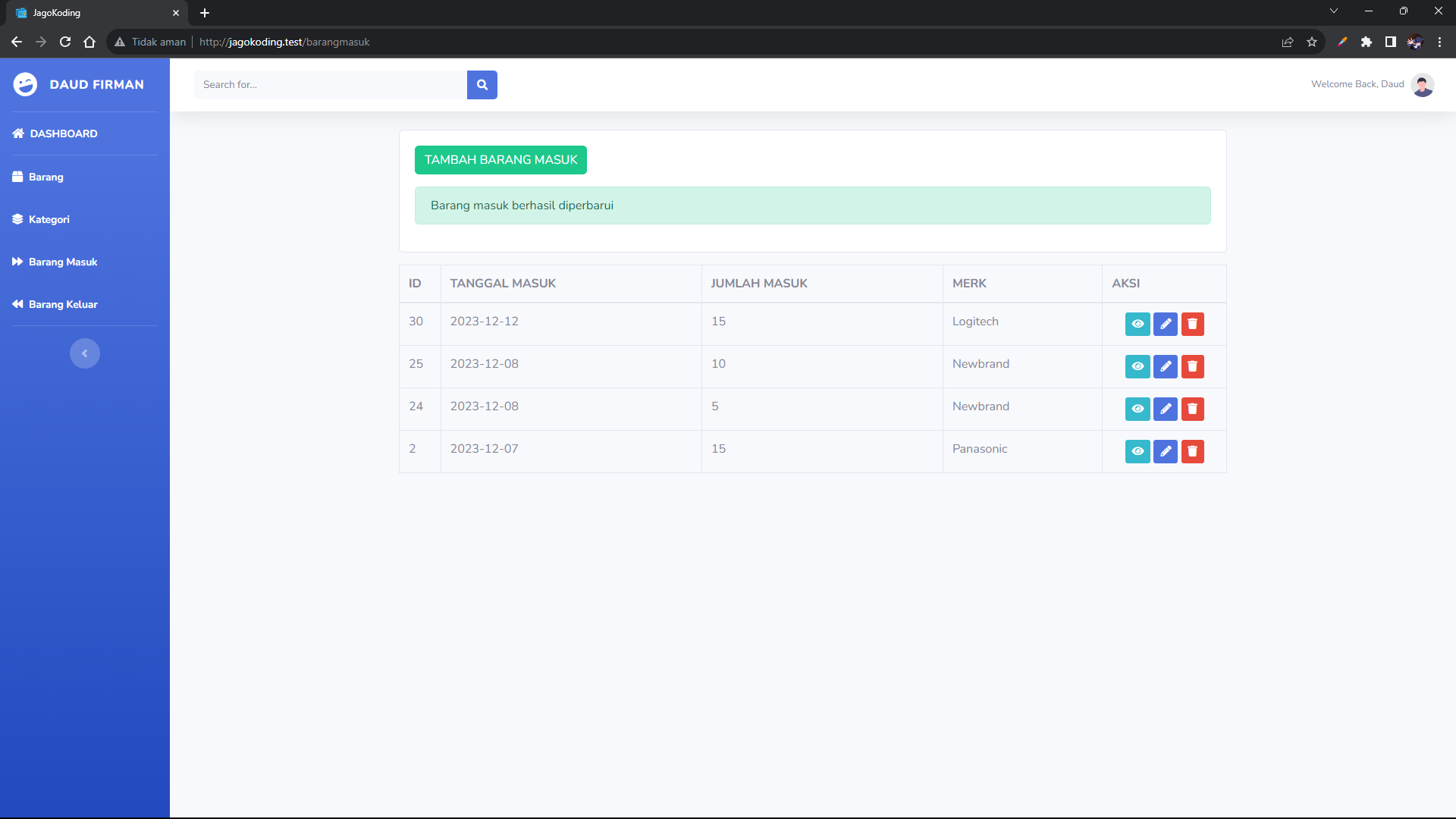


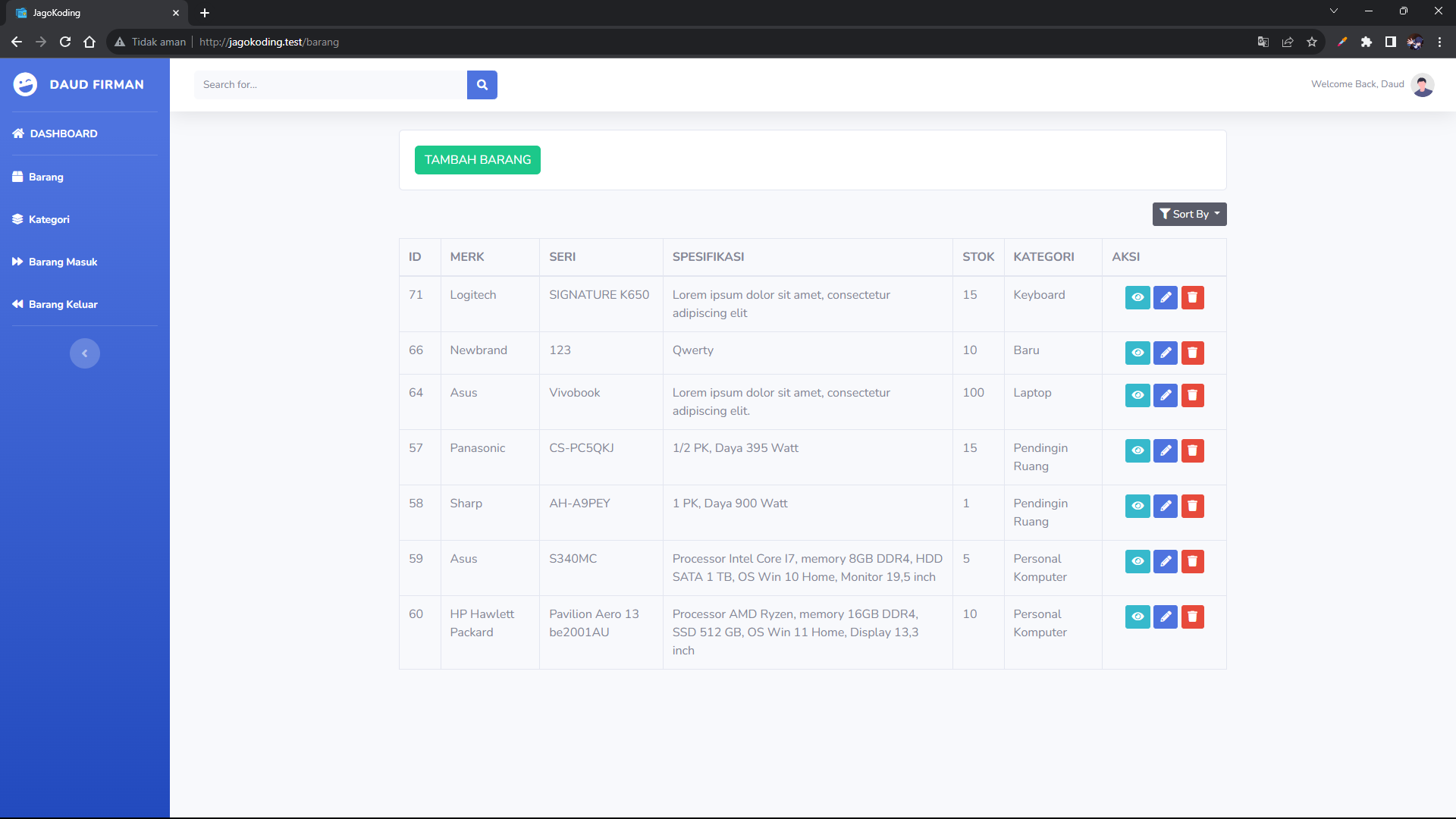




* + Edit

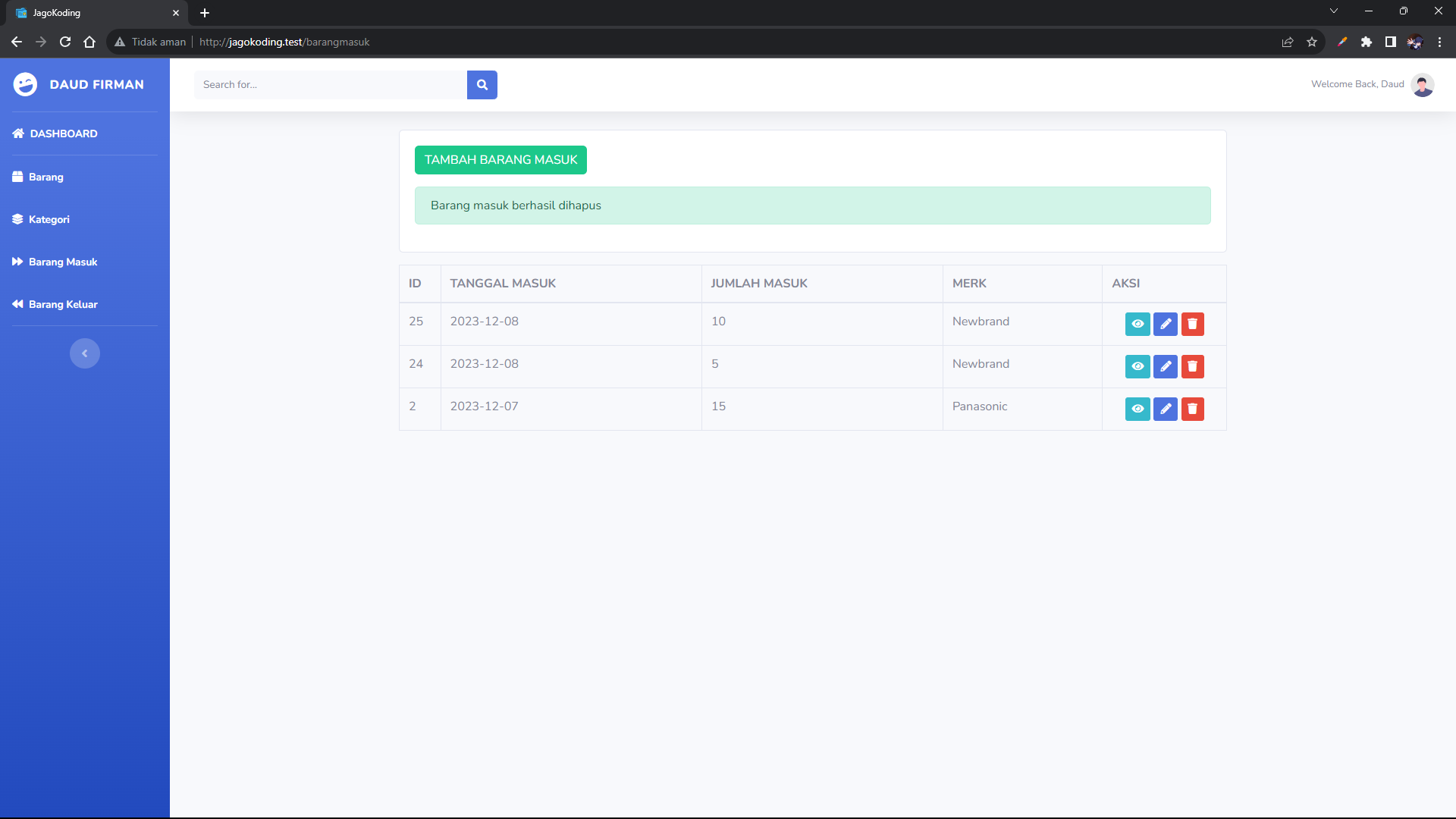


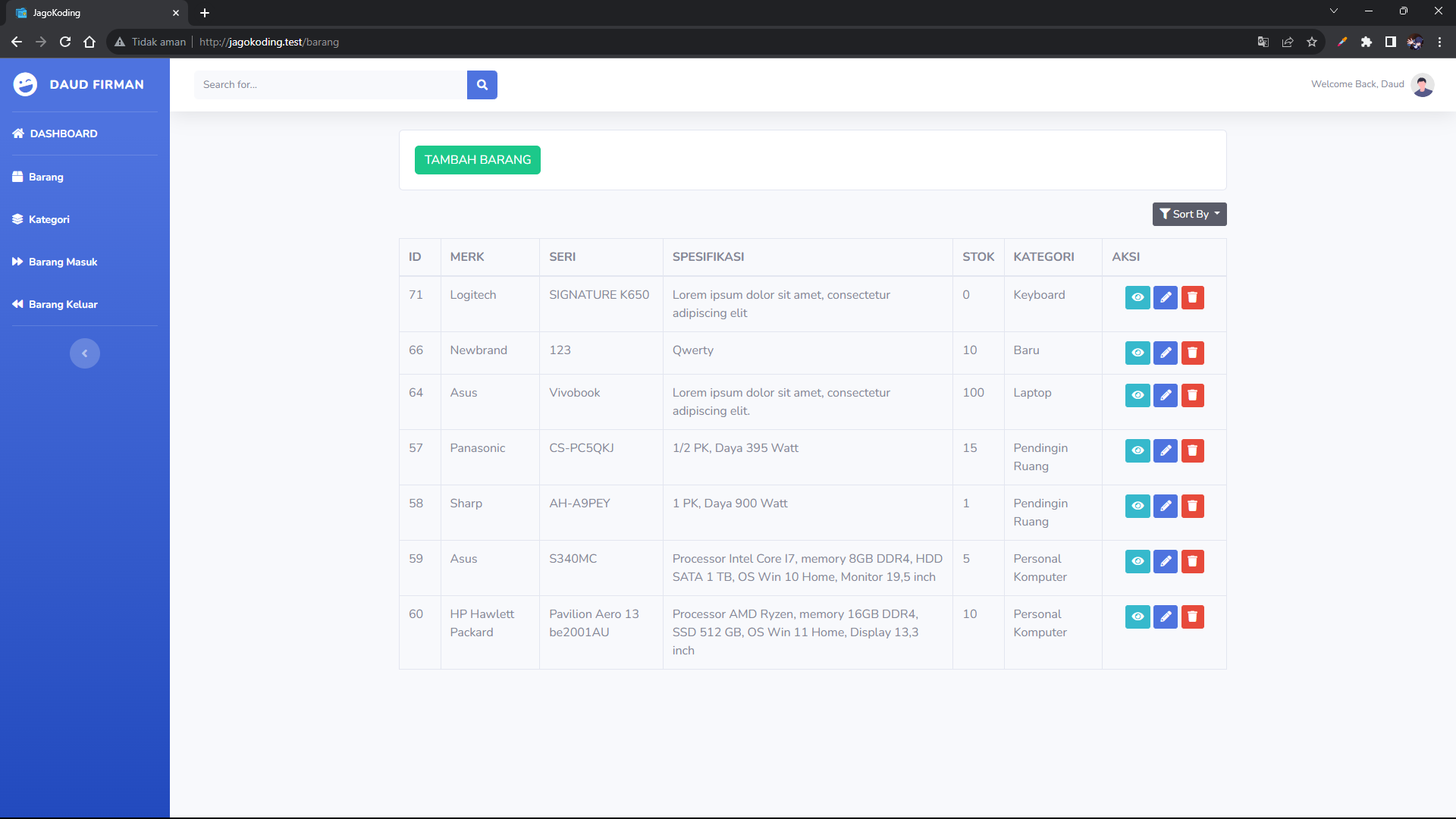




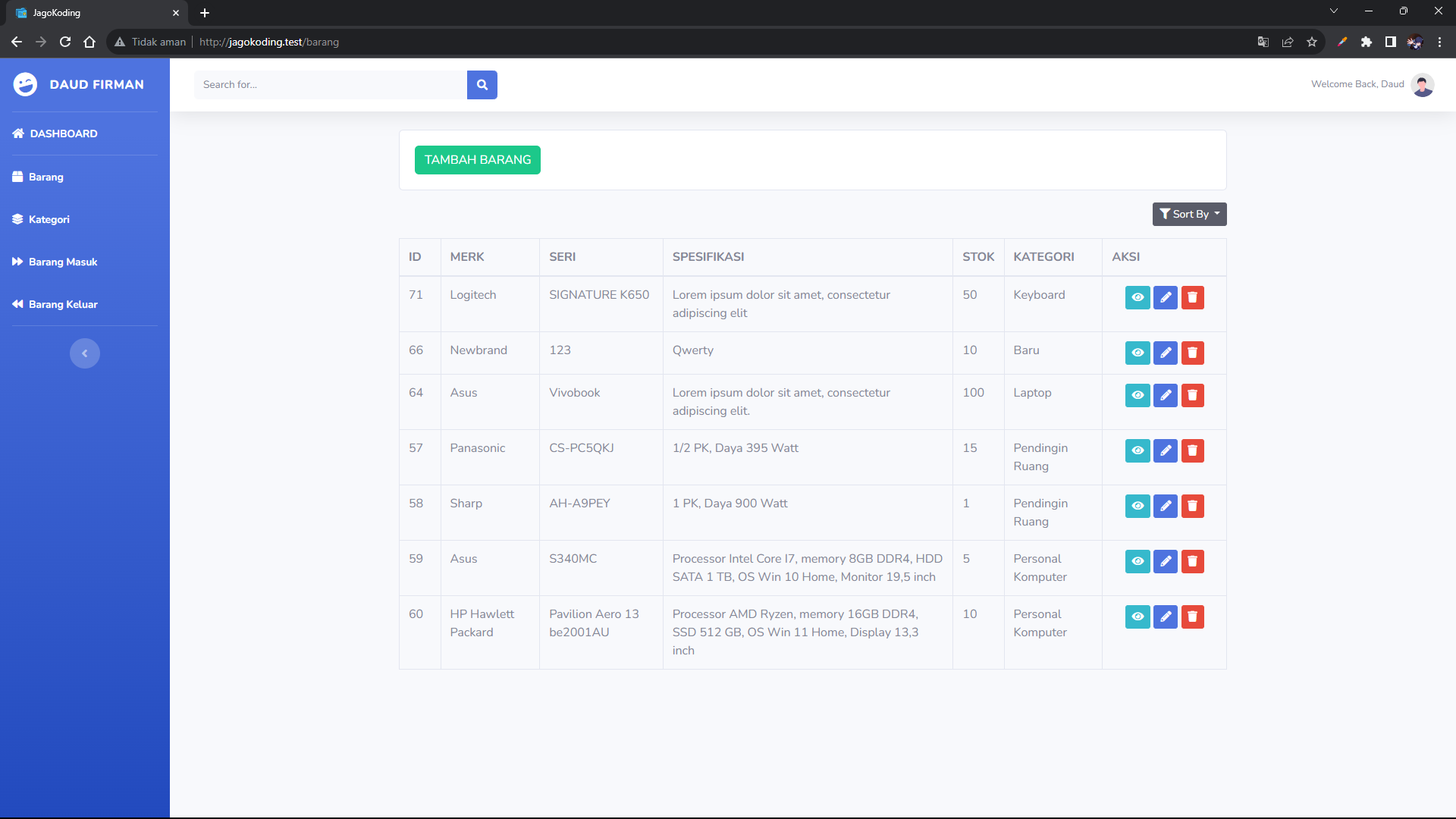
* + Delete



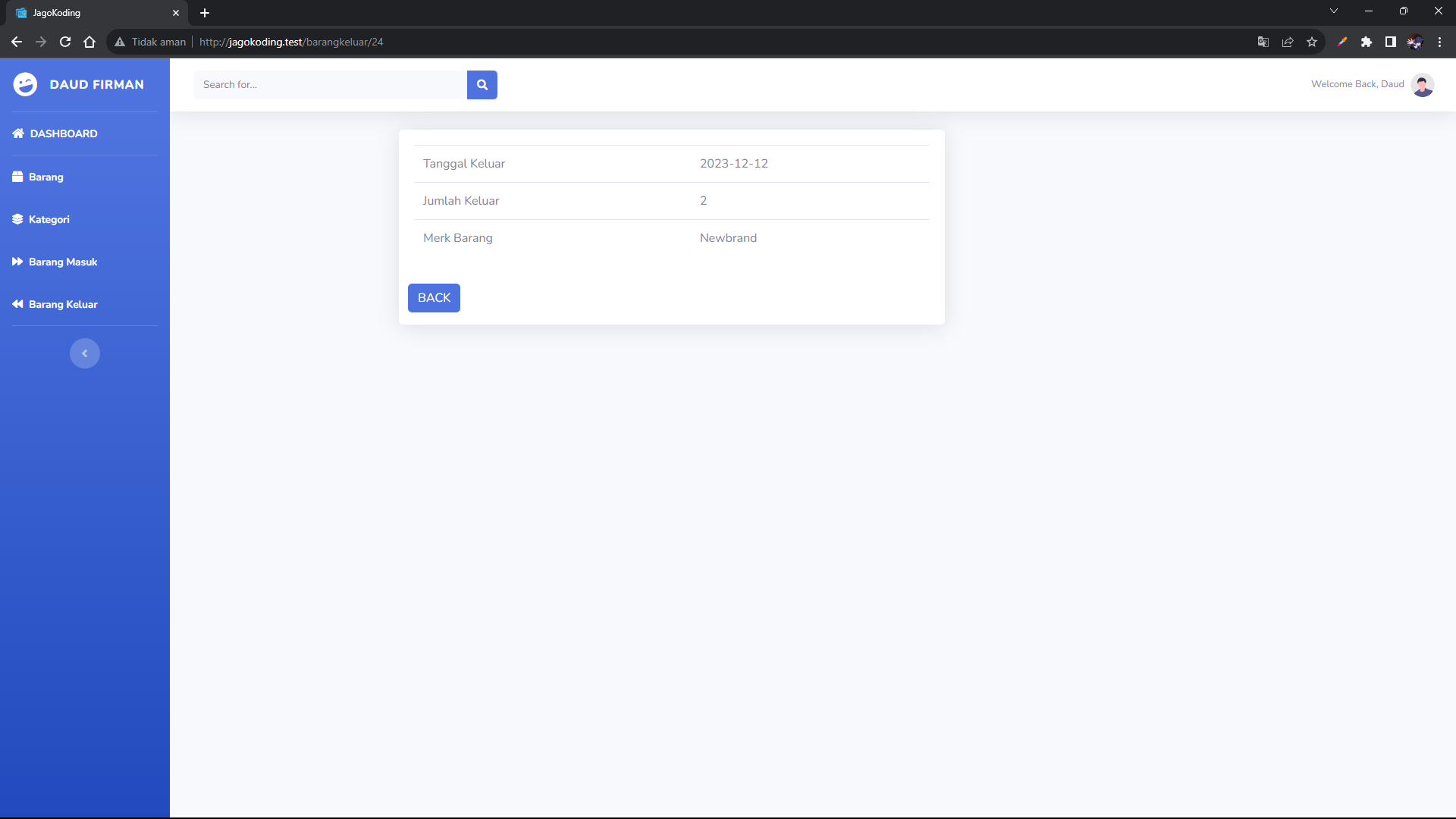




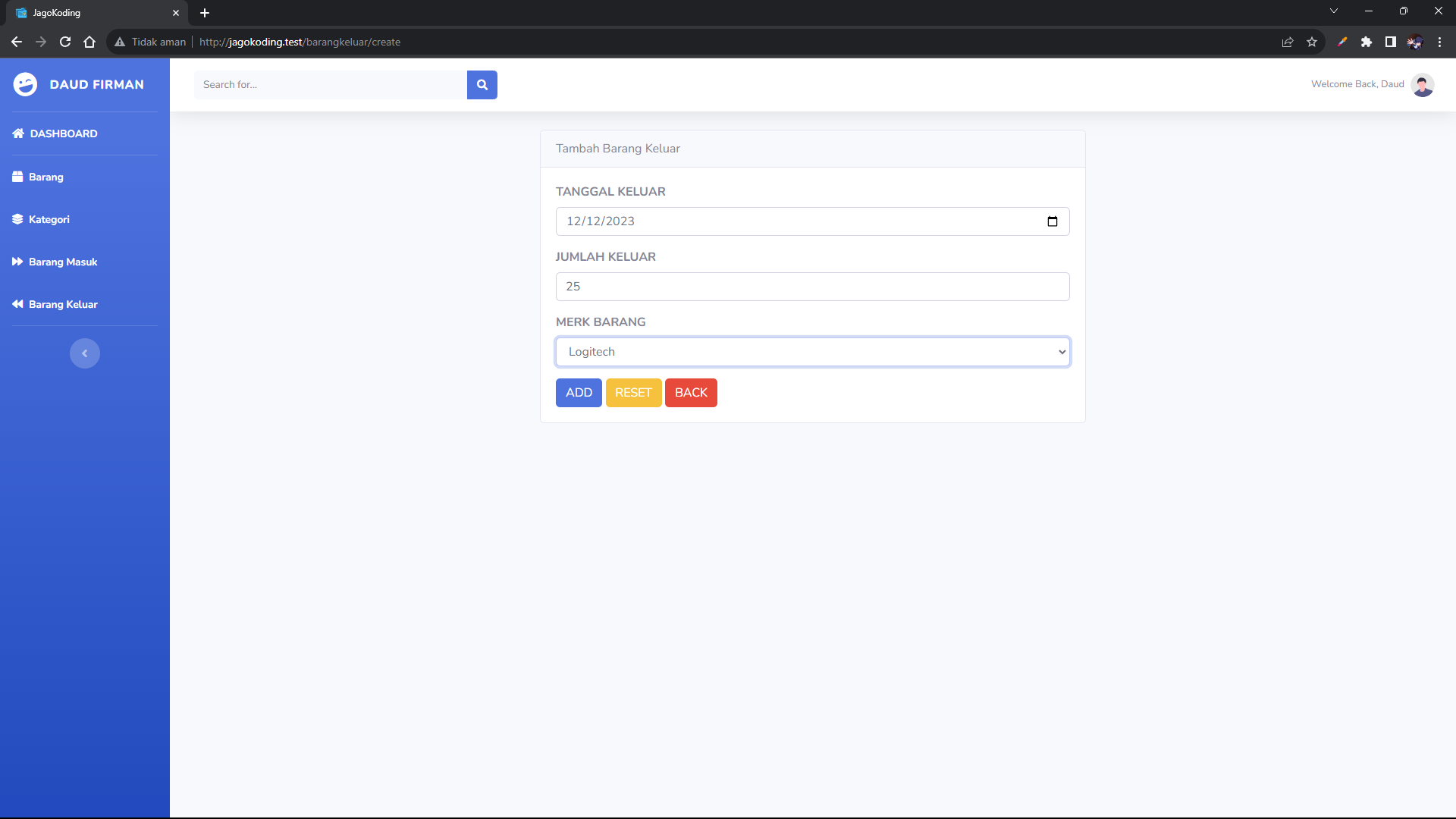
* Halaman Barang Keluar

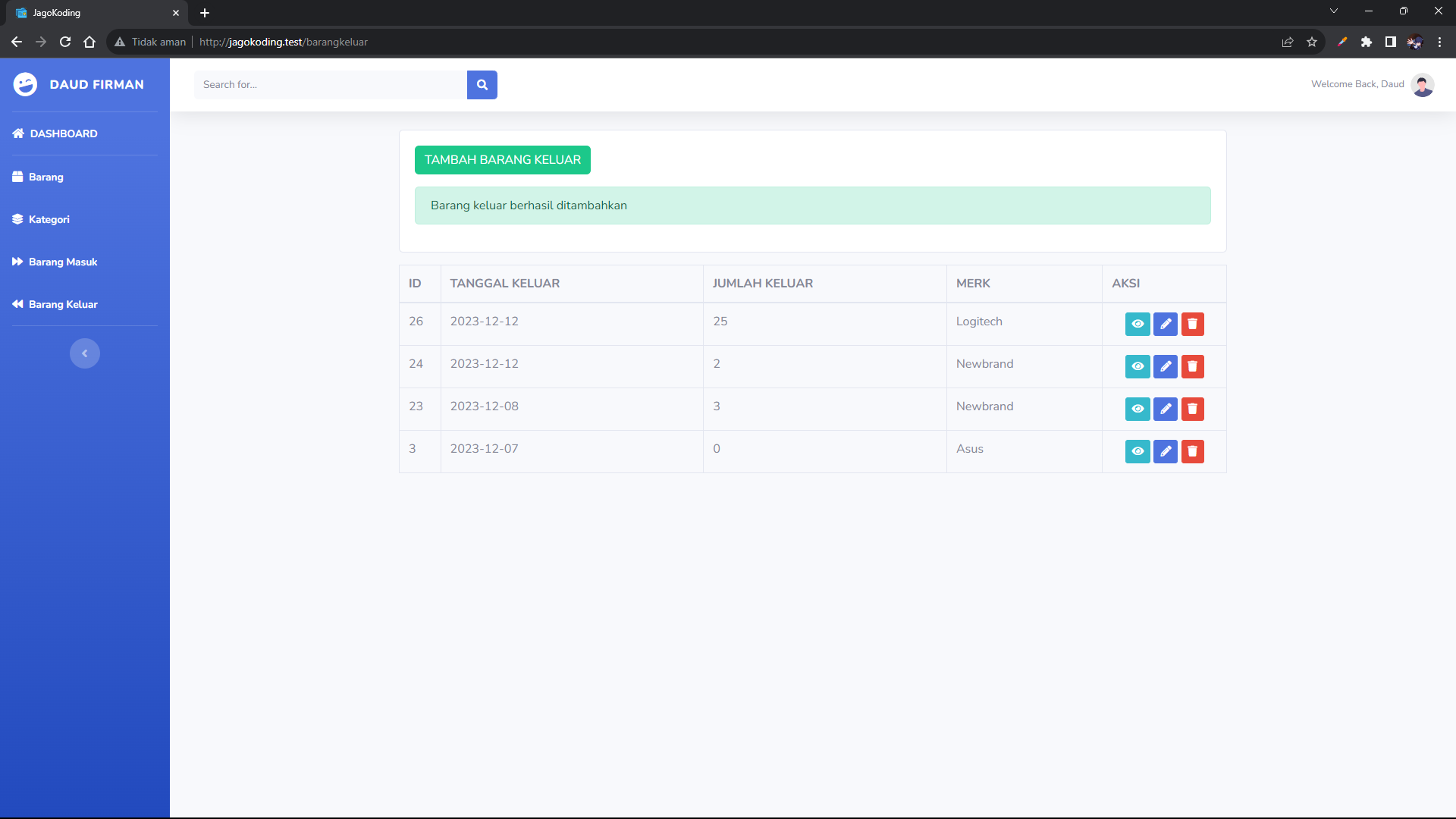


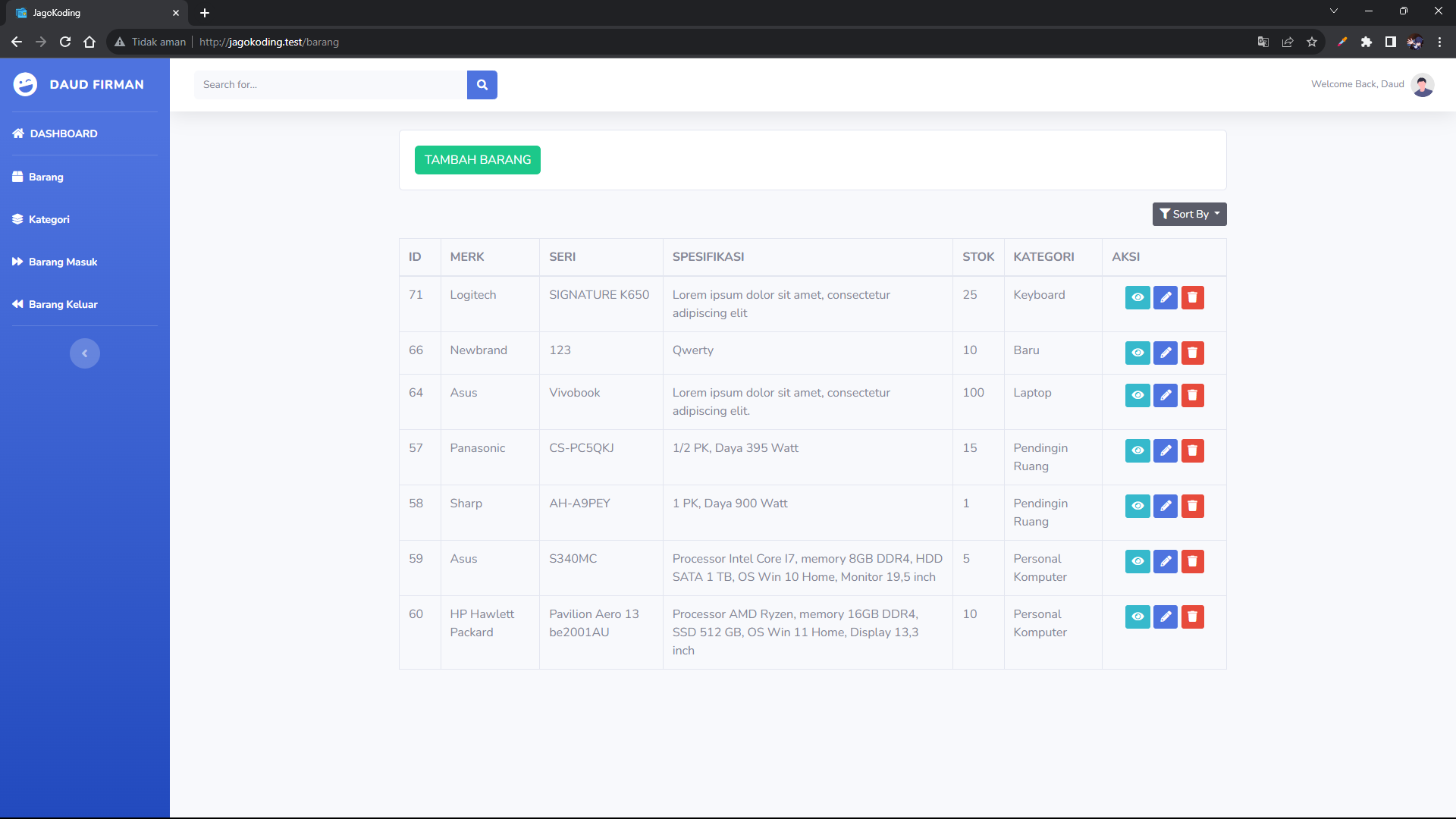
* + Show



* + New

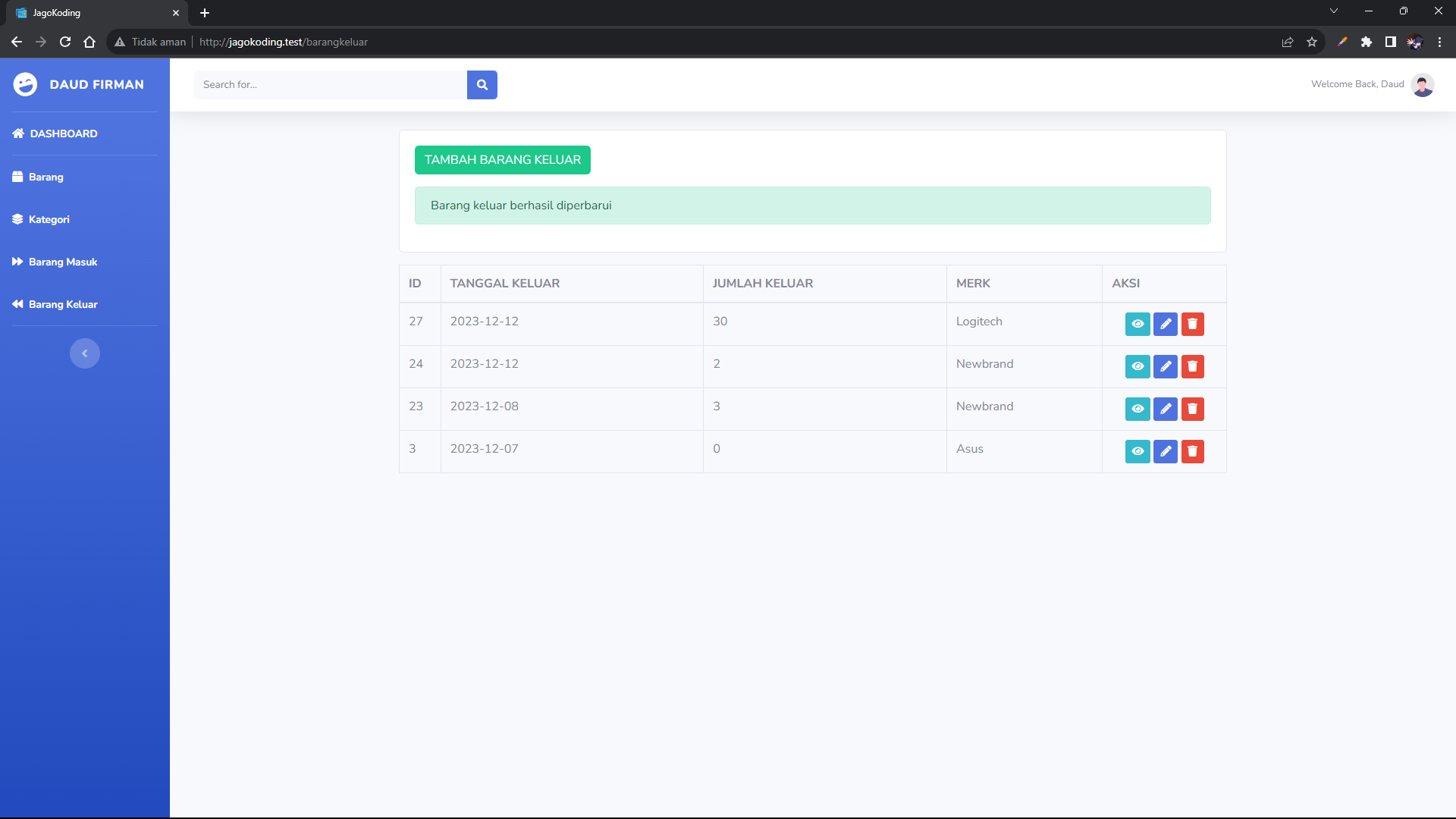


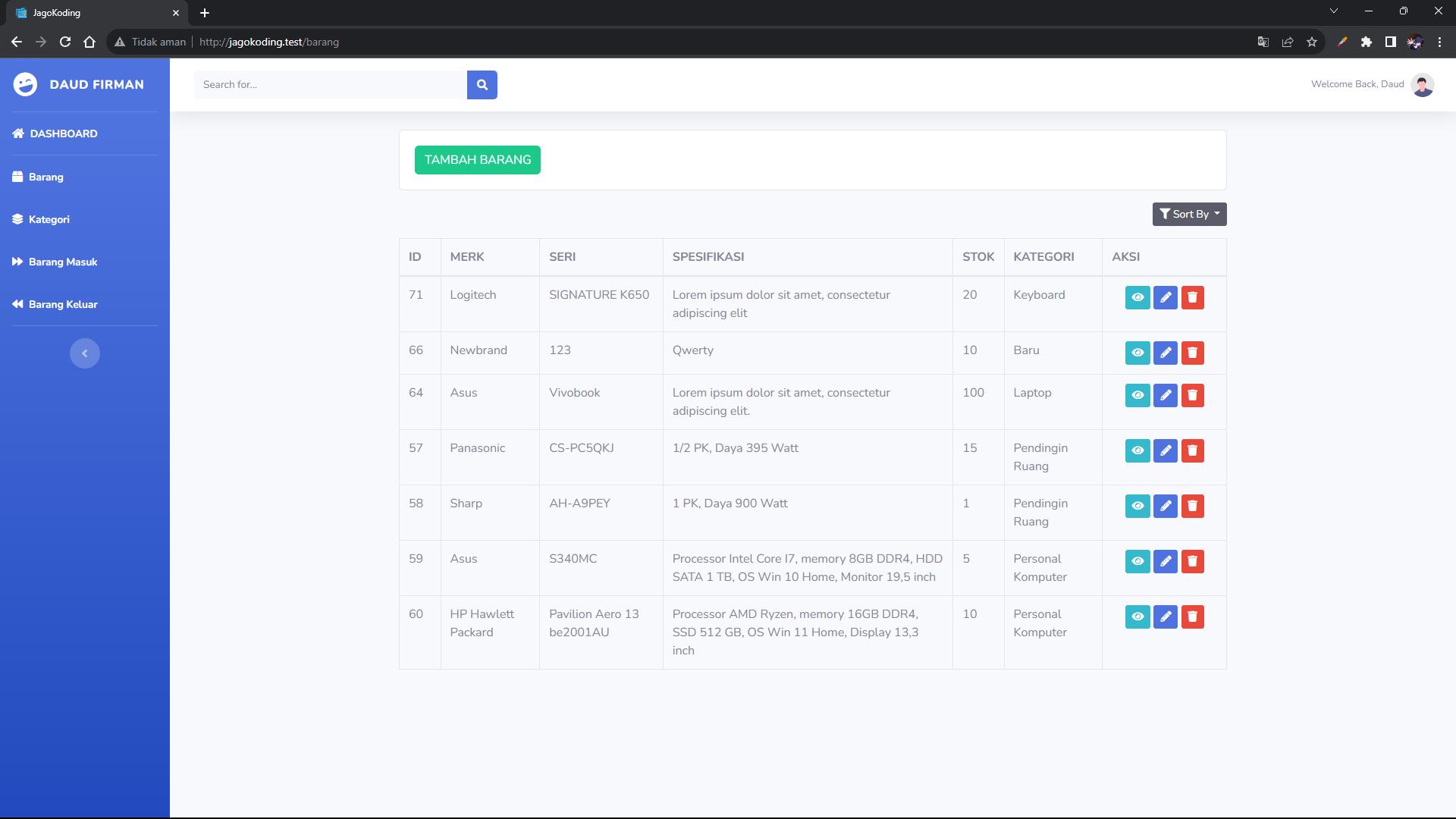




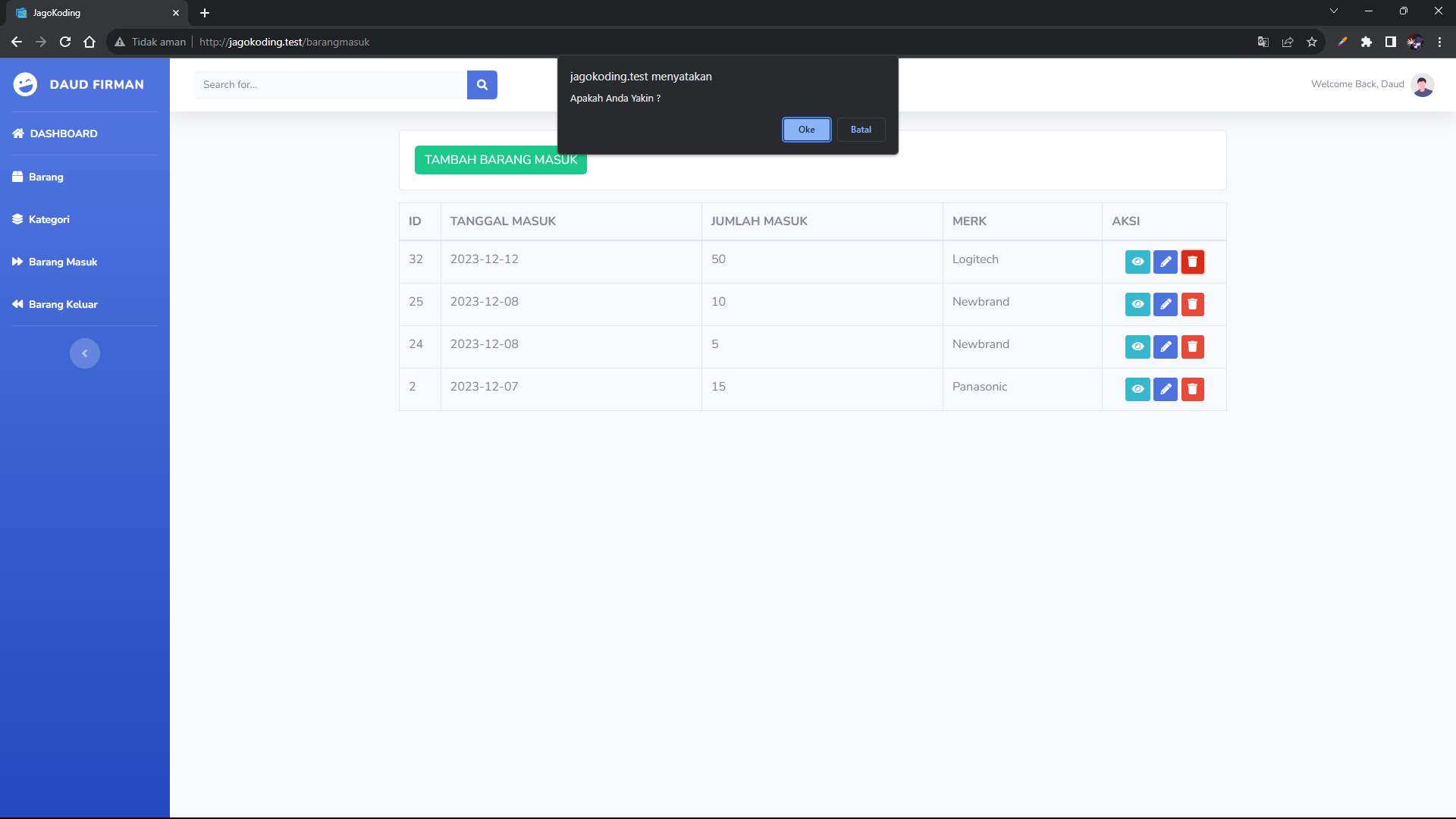
* + Edit

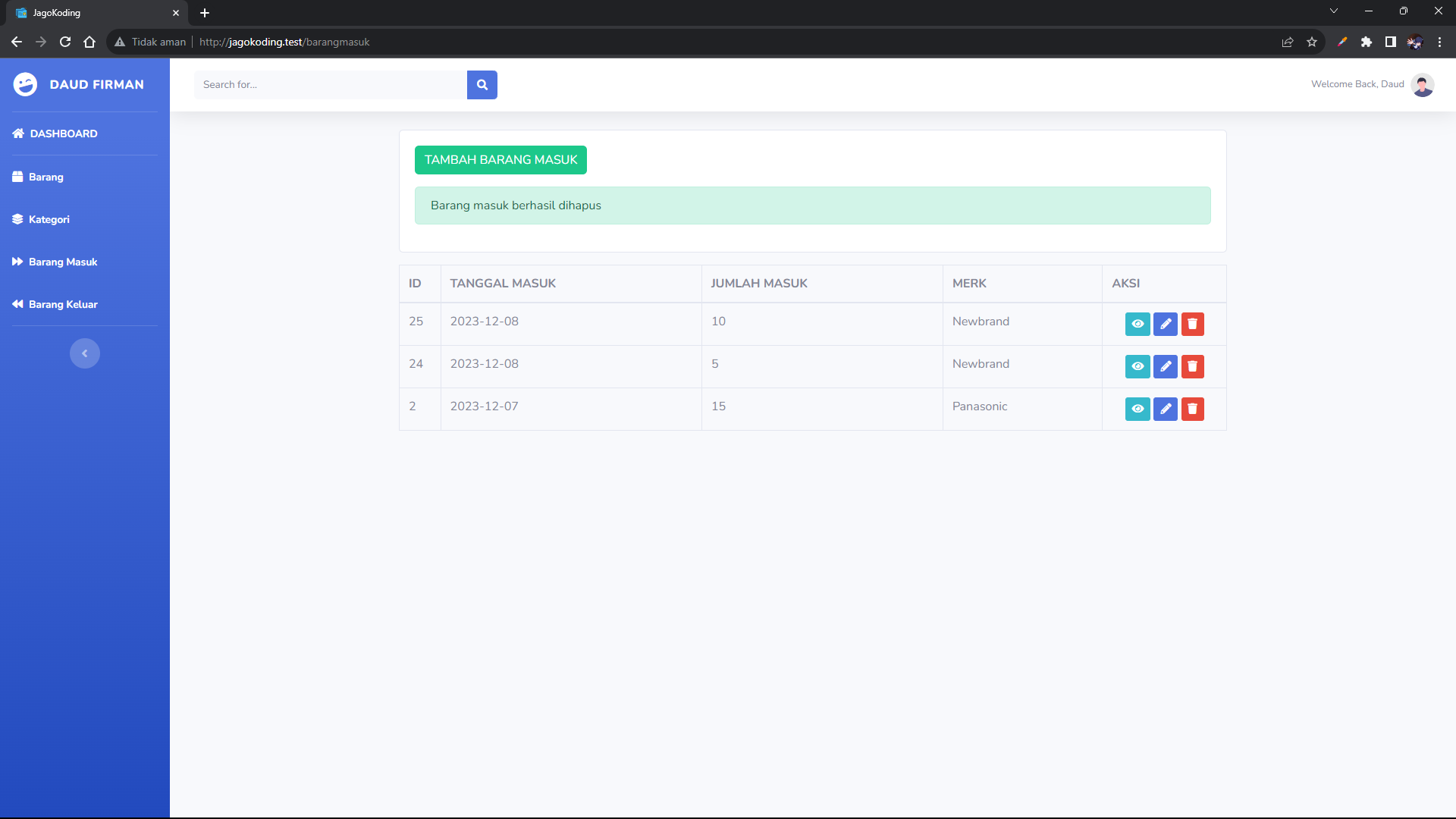


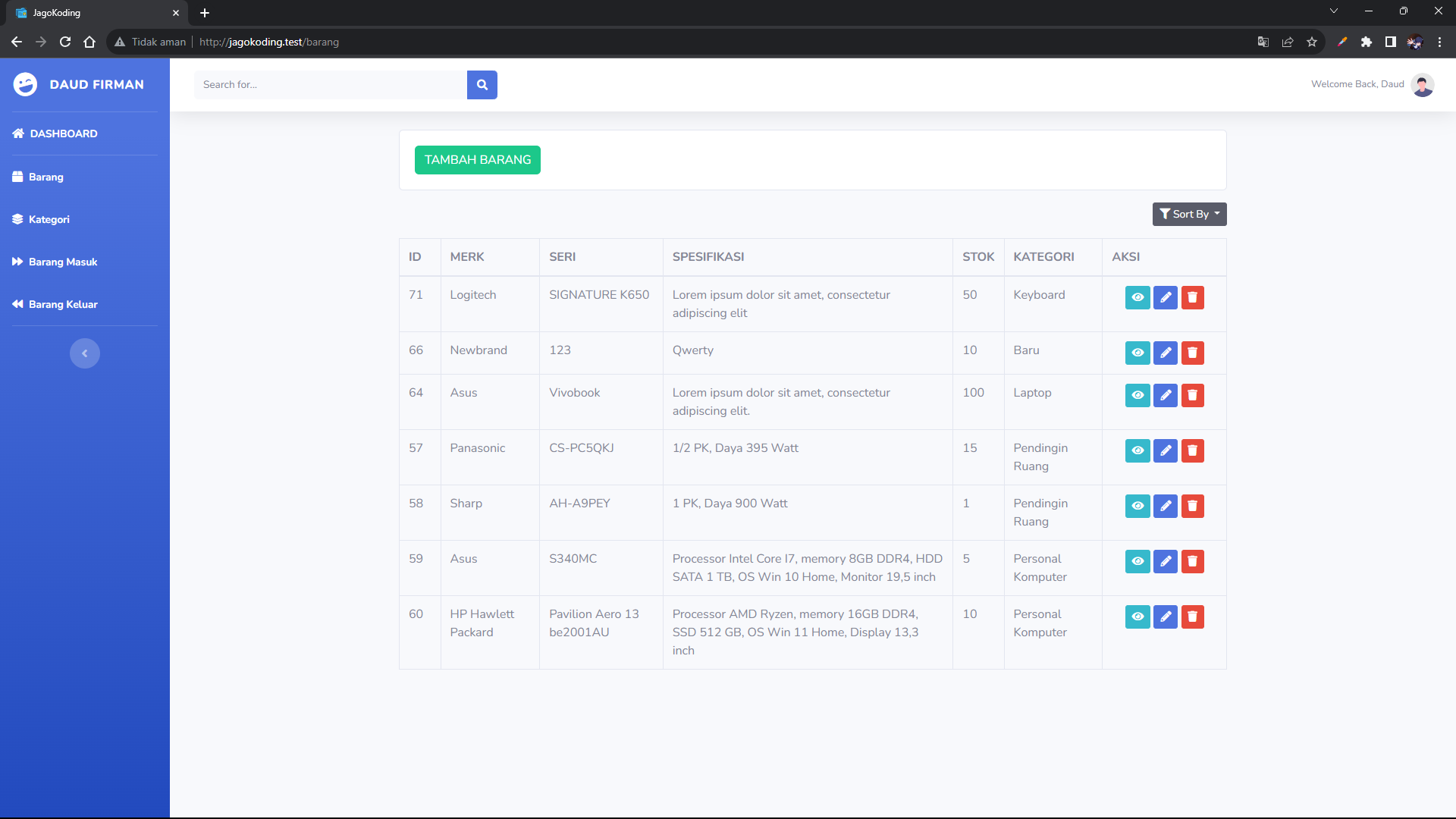




* + Delete







**Source Kode :** [**https://github.com/daudfirmann/inventory-projectlaravel**](https://github.com/daudfirmann/inventory-projectlaravel)