Lab Work 1: User Management System (Registration and Login)

Objective:
Create a basic user authentication system including user registration and login functionality.

Steps:

1. Create Database

For this lab, we used PostgreSQL as the relational database, with NextJS.

- Prisma was used as the ORM to define and interact with the database
- User model in schema.prisma:

```
model User {
  id String @id @default(cuid())
  name String
  email String @unique
  password String
  dob DateTime
  createdAt DateTime @default(now())

  // Relations
  cartItems Cart[]
  reviews Review[]
  orders Order[]
}
```

2. Registration System

The registration logic is defined in a server action (auth.action.ts) and triggered from a form in the client side. The server action validates:

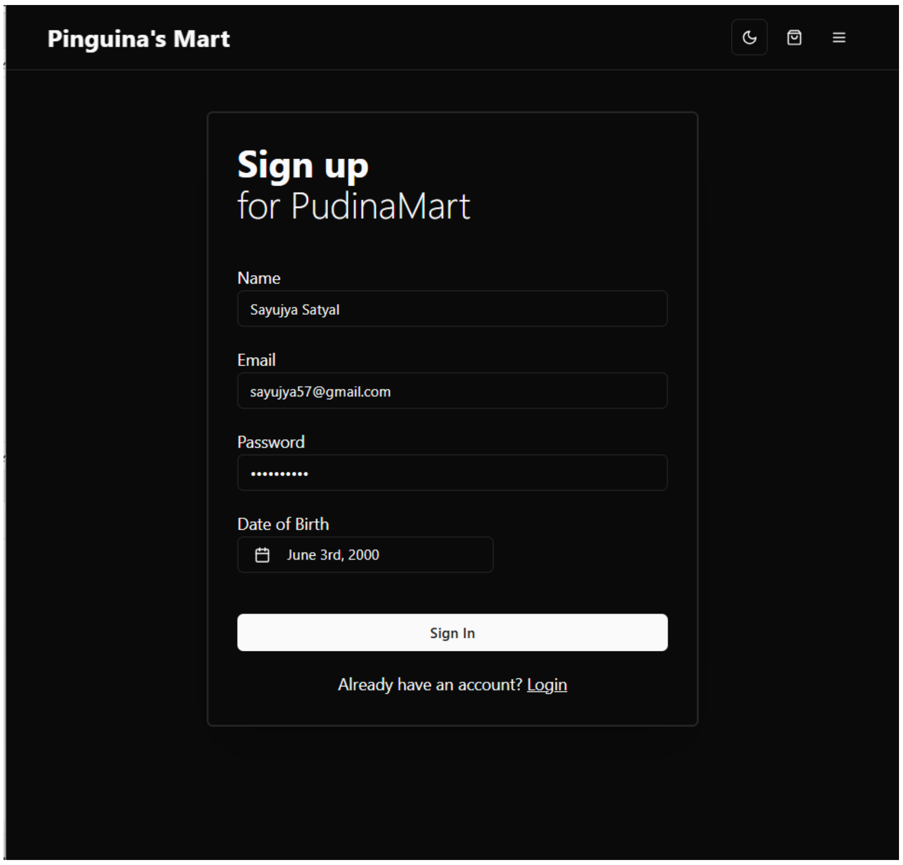
- Required fields
- Age restriction (must be ≥ 13)
- Password strength (using a utility function)
- Uniqueness of email
- If valid, the password is hashed using bcryptjs, and the user is saved in the PostgreSQL database via Prisma.

Source Code:

```
"use server";
import { SignUpFormType } from "@/types/auth.types";
import validatePassword from "@/utils/validatePassword";
import { differenceInYears } from "date-fns";
import bcrypt from "bcryptjs";
import { prisma } from "@/lib/prisma";
```

```
export async function registerUser({
  name,
  email,
  password,
  dob,
}: SignUpFormType) {
  try {
    if (!name || !email || !password || !dob) {
      return { success: false, error: "All Fields are required" };
    }

    const existingUser = await prisma.user.findUnique({
      where: { email },
    });
    if (existingUser) {
      return { success: false, error: "Email is already registered"
};
    }

    const userAge = differenceInYears(new Date(), new Date(dob));
    if (userAge < 13) {
      return {
        success: false,
        error: "You must be at least 13 years old to register.",
      };
    }

    const { isValid, errors } = validatePassword(password);
    if (!isValid) {
      return { success: false, error: errors.join(", ") };
    }

    const hashedPassword: string = await bcrypt.hash(password, 10);

    const newUser = await prisma.user.create({
      data: {
        name: name,
        email: email,
        password: hashedPassword,
        dob: dob || "",
      },
    });

    return {
      success: true,
      user: {
        id: newUser.id,
        name: newUser.name,
        email: newUser.email,
      },
    };
  } catch (error) {
    console.log("Failed to Register User: ", error);
    return { success: false, error: "Failed to Register" };
  }
}
```

Outputs:



Pinguina's Mart

Sign up
for PudinaMart

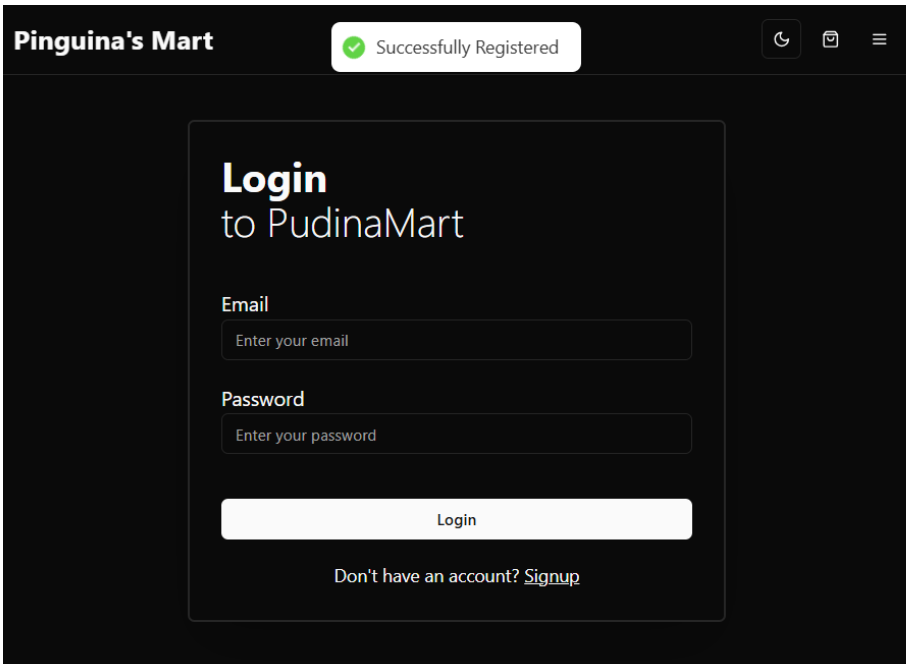
Name: Sayujya Satyal

Email: sayujya57@gmail.com

Password: ••••••••••

Date of Birth: June 3rd, 2000

Figure 1: Registration form for users



Pinguina's Mart

Successfully Registered

Login
to PudinaMart

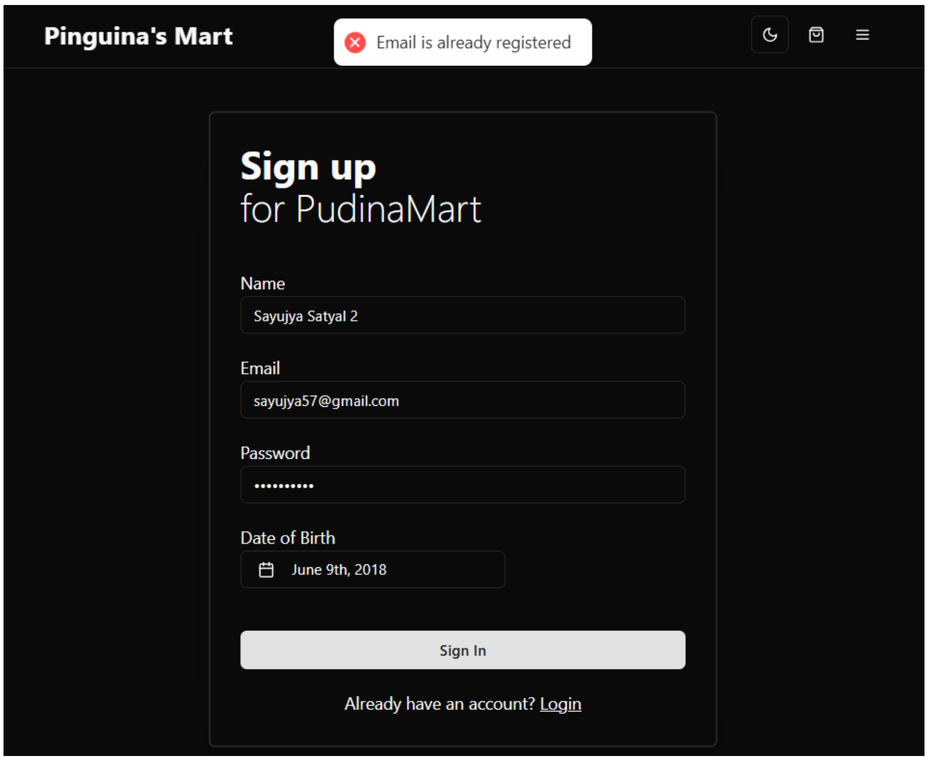
Email
Enter your email

Password
Enter your password

Login

Don't have an account? Signup

Figure 2: Successful registration



Pinguina's Mart

Email is already registered

Sign up
for PudinaMart

Name
Sayujya Satyal 2

Email
sayujya57@gmail.com

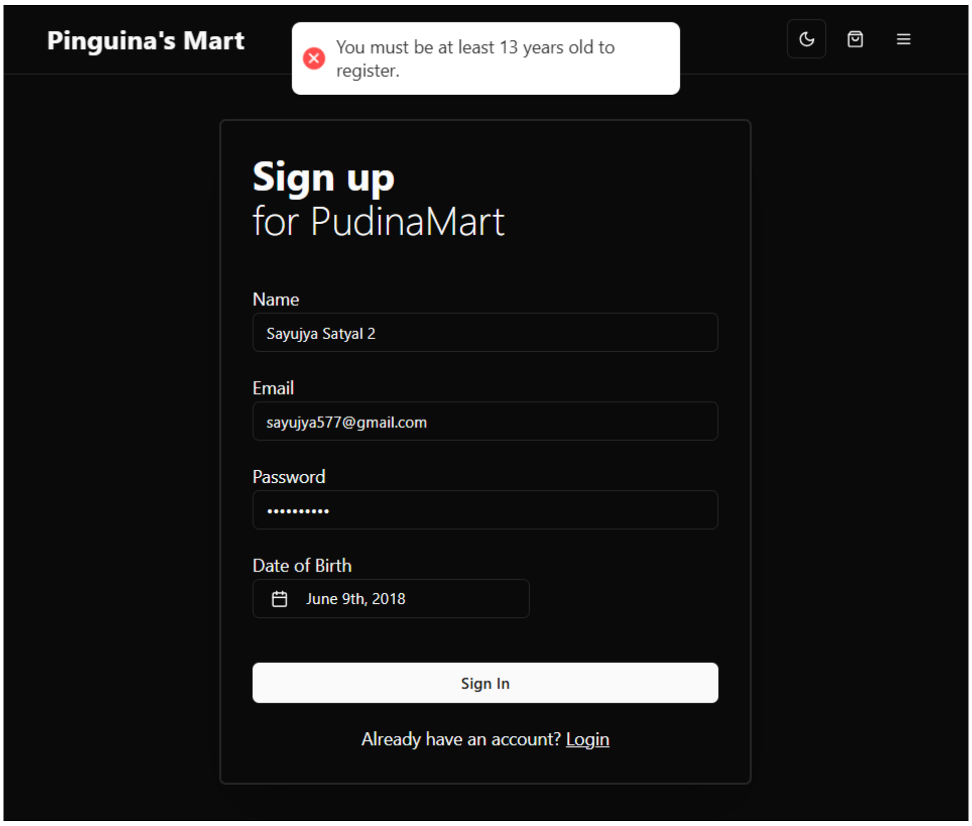
Password
••••••••••

Date of Birth
June 9th, 2018

Sign In

Already have an account? Login

Figure 3: Duplicate email registration


Pinguina's Mart

You must be at least 13 years old to register.

Sign up
for PudinaMart

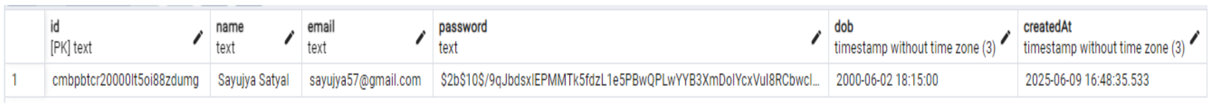
Name: Sayujya Satyal 2

Email: sayujya577@gmail.com

Password: ••••••••••

Date of Birth: June 9th, 2018

Figure 4: Age validation during registration



	id [PK] text	name text	email text	password text	dob timestamp without time zone (3)	createdAt timestamp without time zone (3)
1	cmbpbtcr20000lt5oi88zdumg	Sayujya Satyal	sayujya57@gmail.com	$2b$10$/9qJbdsxlEPMMTk5fdzL1e5PBwQPLwYYB3XmDolYcxVul8RCbwcl...	2000-06-02 18:15:00	2025-06-09 16:48:35.533

Figure 5: Stored data in database after registration

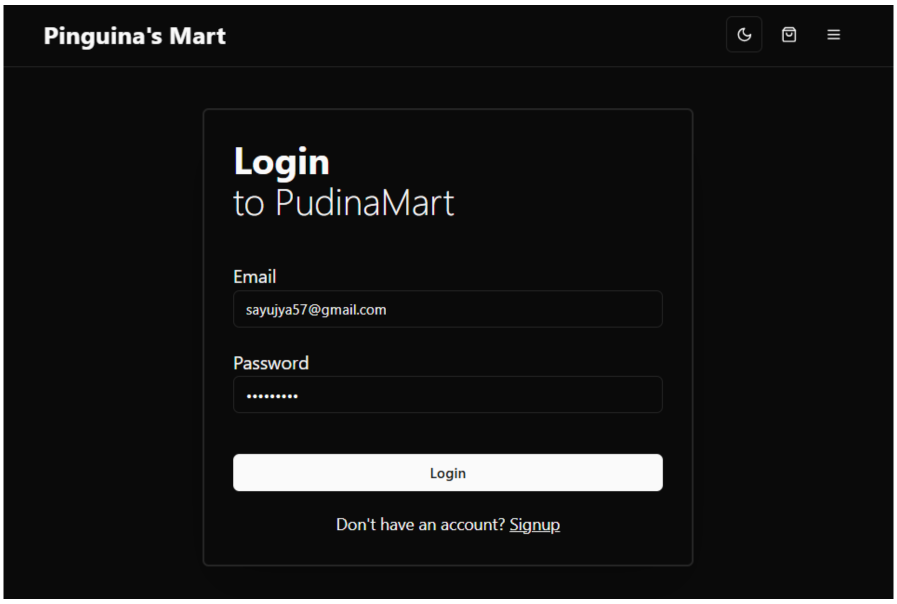
3. Login System:

- The login form accepts email and password and sends a request to the /api/login route.
- On the server:
  - Credentials are validated against stored hashes.
  - A JWT token is issued on success and stored in an HTTP-only cookie.

Source Code:

```
import { NextRequest, NextResponse } from "next/server";
import { login } from "@/lib/auth";
import { prisma } from "@/lib/prisma";
import bcrypt from "bcryptjs";

export async function POST(req: NextRequest) {
  const body = await req.json();
  const { email, password } = body;

  if (!email || !password) {
    return NextResponse.json(
      { error: "Email and password required" },
      { status: 400 }
    );
  }

  const user = await prisma.user.findUnique({ where: { email }
});

  if (!user || !(await bcrypt.compare(password,
user.password))) {
    return NextResponse.json({ error: "Invalid credentials" },
{ status: 401 });
  }

  await login({ id: user.id, email: user.email, name:
user.name });

  return NextResponse.json({
    success: true,
    user: { id: user.id, email: user.email, name: user.name },
  });
}
```

Outputs:



Pinguina's Mart

Login
to PudinaMart

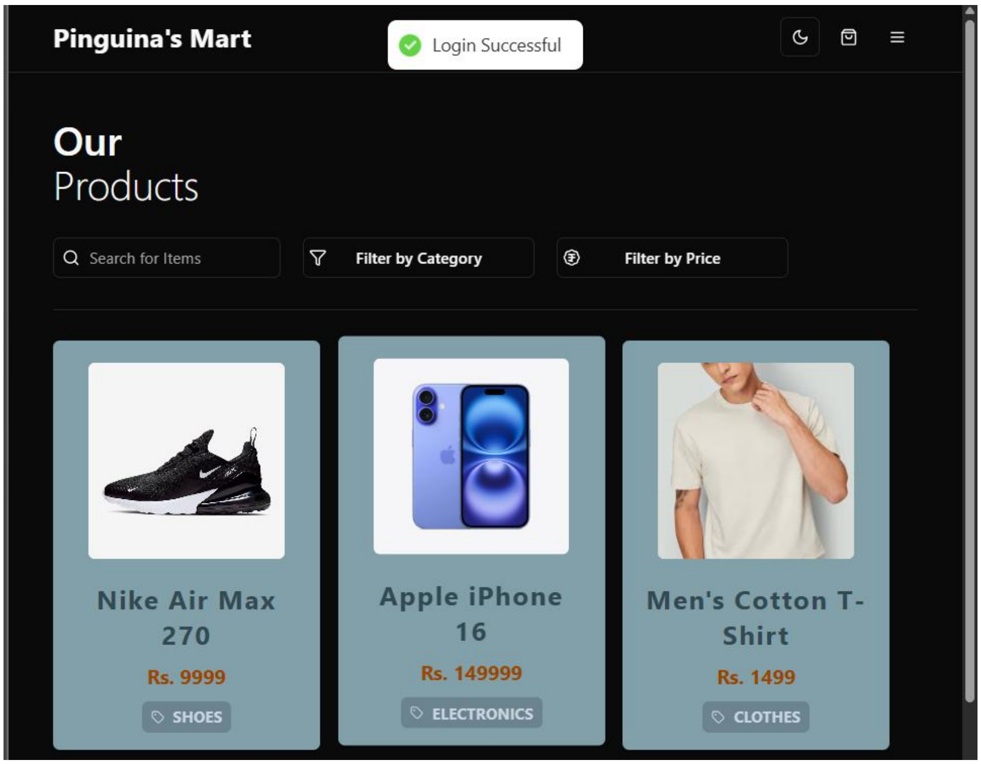
Email
sayujya57@gmail.com

Password
•••••••••

Login

Don't have an account? Signup

Figure 6: Login form


Pinguina's Mart

Login Successful

Our
Products

Search for Items

Filter by Category

Filter by Price

Nike Air Max 270

Rs. 9999

SHOES

Apple iPhone 16

Rs. 149999

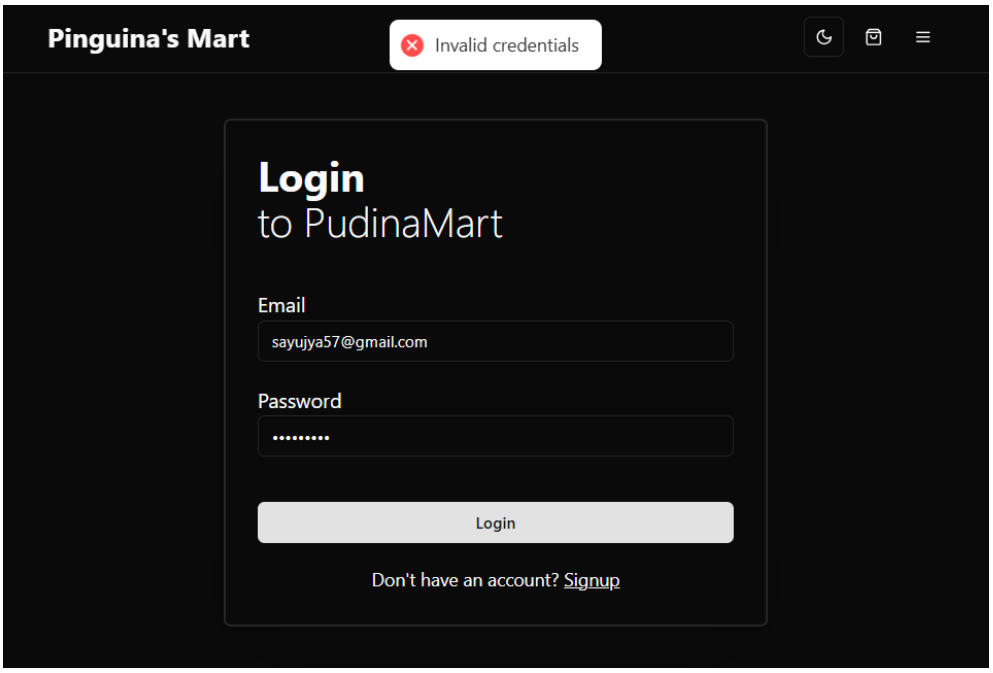
ELECTRONICS

Men's Cotton T-Shirt

Rs. 1499

CLOTHES

Figure 7: Successful login


Pinguina's Mart

Invalid credentials

Login
to PudinaMart

Email

sayujya57@gmail.com

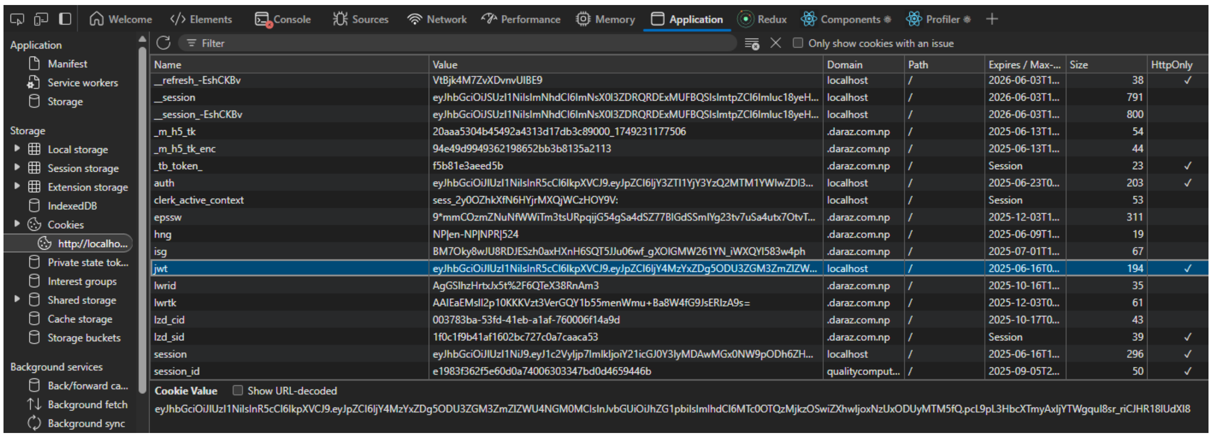
Password

•••••••••

Login

Don't have an account? Signup

Figure 8: Invalid credentials using login



Name	Value	Domain	Path	Expires / Max-...	Size	HttpOnly
__refresh_-EshCKBv	VtBjk4M7ZvXDvnvUIBE9	localhost	/	2026-06-03T1...	38	✓
__session	eyJhbGciOiJSUzI1NiIsImNhdCI6ImNsX0I3ZDRQRDExMUFBQSIsImtpZCI6Imluc18yeH...	localhost	/	2026-06-03T1...	791	
__session_-EshCKBv	eyJhbGciOiJSUzI1NiIsImNhdCI6ImNsX0I3ZDRQRDExMUFBQSIsImtpZCI6Imluc18yeH...	localhost	/	2026-06-03T1...	800	
_m_h5_tk	20aaa5304b45492a4313d17db3c89000_1749231177506	.daraz.com.np	/	2025-06-13T1...	54	
_m_h5_tk_enc	94e49d9949362198652bb3b8135a2113	.daraz.com.np	/	2025-06-13T1...	44	
_tb_token_	f5b81e3aeed5b	.daraz.com.np	/	Session	23	✓
auth	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3ZTI1YjY3YzQ2MTM1YWIwZDI3...	localhost	/	2025-06-23T0...	203	✓
clerk_active_context	sess_2y0OZhkXfN6HYjrMXQjWCzHOY9V:	localhost	/	Session	53	
epssw	9*mmCOzmZNuNfWWiTm3tsURpqijG54gSa4dSZ77BIGdSSmIYg23tv7uSa4utx7OtvT...	.daraz.com.np	/	2025-12-03T1...	311	
hng	NP|en-NP|NPR|524	.daraz.com.np	/	2025-06-09T1...	19	
isg	BM7Oky8wJU8RDJESzh0axHXnH6SQT5JJu06wf_gXOIGMW261YN_iWXQYI583w4ph	.daraz.com.np	/	2025-07-01T1...	67	
jwt	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY4MzYxZDg5ODU3ZGM3ZmZlZW...	localhost	/	2025-06-16T0...	194	✓
lwrid	AgGSIhzHrtxJx5t%2F6QTeX38RnAm3	.daraz.com.np	/	2025-10-16T1...	35	
lwrtk	AAIEaEMsII2p10KKKVzt3VerGQY1b55menWmu+Ba8W4fG9JsERIzA9s=	.daraz.com.np	/	2025-12-03T0...	61	
lzd_cid	003783ba-53fd-41eb-a1af-760006f14a9d	.daraz.com.np	/	2025-10-17T0...	43	
lzd_sid	1f0c1f9b41af1602bc727c0a7caaca53	.daraz.com.np	/	Session	39	✓
session	eyJhbGciOiJIUzI1NiJ9.eyJ1c2VyIjp7ImlkIjoiY21icGJ0Y3lyMDAwMGx0NW9pODh6ZH...	localhost	/	2025-06-16T1...	296	✓
session_id	e1983f362f5e60d0a74006303347bd0d4659446b	qualitycomput...	/	2025-09-05T2...	50	✓

Cookie Value ☐ Show URL-decoded

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY4MzYxZDg5ODU3ZGM3ZmZlZWU4NGM0MCIsInJvbGUiOiJhZG1pbiIsImlhdCI6MTc0OTQzMjkzOSwiZXhwIjoxNzUxODUyMTM5fQ.pcL9pL3HbcXTmyAxIjYTWgqul8sr_riCJHR18IUdXI8
```

Figure 9: JWT Cookie after successful Login

Results

A secure registration and login system that:

- o Validates age, password, and uniqueness of email.
- o Hashes passwords before saving.
- o Authenticates users and issues a JWT for session management.
- o Redirects authenticated users to a dashboard.