# **Name: Abdur R. Qureshi**

# **Roll No: 242466**

Assignment No: 2

Date Of Performance: 10/10/2025

Q: Implementation of Nodejs/ Express Rest API applications.

CODE (src/app/api/\*):

==> delete/pinata/delete-file/route.ts

import { NextResponse } from "next/server";

import { pinata } from "@/lib/pinata/config";

export async function DELETE(request: Request) {

try {

const { id } = await request.json();

if (!id) {

return NextResponse.json(

{ error: "File ID is required" },

{ status: 400 }

);

}

const deletedFiles = await pinata.files.public.delete([id]);

return NextResponse.json(deletedFiles, { status: 200 });

} catch (e) {

console.error("Server error:", e);

return NextResponse.json(

{ error: "Internal server error" },

{ status: 500 }

);

}

}

==> get/neon/orders/fetch-all/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

export async function GET() {

try {

const query = `

SELECT \* FROM "orders"

ORDER BY ordered\_at DESC;

`;

const result = await pool.query(query);

return NextResponse.json({

data: result.rows,

status: true,

});

} catch (error) {

console.error("Error fetching all orders:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> get/neon/orders/in-cart/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

export async function GET() {

try {

const query = `

SELECT \* FROM "orders"

WHERE in\_cart = true

ORDER BY ordered\_at DESC;

`;

const result = await pool.query(query);

return NextResponse.json({

data: result.rows,

status: true,

});

} catch (error) {

console.error("Error fetching cart orders:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> get/neon/orders/pending-orders/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

export async function GET() {

try {

const query = `

SELECT \* FROM "orders"

WHERE order\_status = 'pending'

ORDER BY ordered\_at ASC;

`;

const result = await pool.query(query);

return NextResponse.json({

data: result.rows,

status: true,

});

} catch (error) {

console.error("Error fetching pending orders:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> get/neon/prints/fetch-all/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

export async function GET() {

try {

const result = await pool.query(

`SELECT \* FROM "prints" ORDER BY uploaded\_at ASC;`

);

return NextResponse.json({ data: result.rows, status: true });

} catch (error) {

console.error("Error fetching all prints:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> get/neon/prints/todays-queue/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

import { PrintRecord } from "@/interfaces/Print";

export async function GET() {

try {

const query = `

SELECT \* FROM "prints"

WHERE print\_status = $1

ORDER BY uploaded\_at DESC;

`;

const result = await pool.query<PrintRecord>(query, ["pending"]);

return NextResponse.json({ data: result.rows, status: true });

} catch (error) {

console.error("Error fetching queue:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> patch/neon/orders/cancel-order/route.ts

import pool from "@/lib/neon/config";

import { NextResponse } from "next/server";

export async function PATCH(req: Request) {

try {

const body = await req.json();

const { order\_id } = body;

if (!order\_id) {

return NextResponse.json(

{

error: "Order ID is required to cancel order",

status: false,

},

{ status: 400 }

);

}

const query = `

UPDATE "orders"

SET

order\_status = 'cancelled'

WHERE order\_id = $1

RETURNING \*;

`;

const values = [order\_id];

const result = await pool.query(query, values);

if (result.rows.length 0) {

return NextResponse.json(

{ error: "Order not found", status: false },

{ status: 404 }

);

}

return NextResponse.json({

data: result.rows[0],

message: "Order cancelled successfully",

status: true,

});

} catch (error) {

console.error("Error cancelling order:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> patch/neon/orders/checkout-order/route.ts

import pool from "@/lib/neon/config";

import { NextResponse } from "next/server";

export async function PATCH(req: Request) {

try {

file.print\_count,

file.page\_count,

file.print\_type,

file.print\_color,

file.binding\_type,

file.instructions || "",

"pending",

getFormatDate(new Date()),

];

const result = await pool.query(query, values);

return NextResponse.json({ data: result.rows, status: true });

} catch (error) {

console.error("Error inserting:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> post/neon/prints/user-history/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

export async function POST(req: Request) {

try {

const { user } = await req.json();

const query = `

SELECT \* FROM "prints"

WHERE user\_id = $1

ORDER BY uploaded\_at ASC;

`;

const result = await pool.query(query, [user.id]);

return NextResponse.json({ data: result.rows, status: true });

} catch (error) {

console.error("Error fetching user history:", error);

return NextResponse.json(

{ error: String(error), status: false },

{ status: 500 }

);

}

}

==> post/pinata/update-file/route.ts

import { NextResponse, type NextRequest } from "next/server";

import { pinata } from "@/lib/pinata/config";

export async function POST(request: NextRequest) {

try {

const { id, file\_name } = await request.json();

if (!id || !file\_name) {

return NextResponse.json(

{ error: "No file provided" },

{ status: 400 }

);

}

const result = await pinata.files.public.update({

id: id,

name: file\_name,

});

return NextResponse.json(

{

id: result.id,

name: result.name,

ipfs\_cid: result.cid,

},

{ status: 200 }

);

} catch (e) {

console.error("Pinata upload error:", e);

return NextResponse.json(

{ error: "Internal Server Error" },

{ status: 500 }

);

}

}

==> post/pinata/upload-files/route.ts

import { NextResponse, type NextRequest } from "next/server";

import { pinata } from "@/lib/pinata/config";

export async function POST(request: NextRequest) {

try {

const data = await request.formData();

const file: File | null = data.get("file") as unknown as File;

if (!file) {

return NextResponse.json(

{ error: "No file provided" },

{ status: 400 }

);

}

const result = await pinata.upload.public.file(file);

const url = await pinata.gateways.public.convert(result.cid);

return NextResponse.json(

{

id: result.id,

ipfs\_hash: result.cid,

ipfs\_url: url,

},

{ status: 200 }

);

} catch (e) {

console.error("Pinata upload error:", e);

return NextResponse.json(

{ error: "Internal Server Error" },

{ status: 500 }

);

}

}

==> put/neon/prints/update-print/route.ts

import { NextResponse } from "next/server";

import pool from "@/lib/neon/config";

export async function PUT(req: Request) {

try {

const { document } = await req.json();

const query = `

UPDATE "prints"

SET

page\_count = $1,

print\_count = $2,

print\_type = $3,

print\_color = $4,

binding\_type = $5,

instructions = $6

WHERE print\_id = $7

RETURNING \*;

`;

const values = [

document.page\_count,

document.print\_count,

document.print\_type,

document.print\_color,

document.binding\_type,

document.instructions,

document.print\_id,

];

const result = await pool.query(query, values);

return NextResponse.json({ data: result.rows[0], status: true });

} catch (error) {

console.error("Error updating:", error);

return NextResponse.json(

{ error: String(error), status: false },

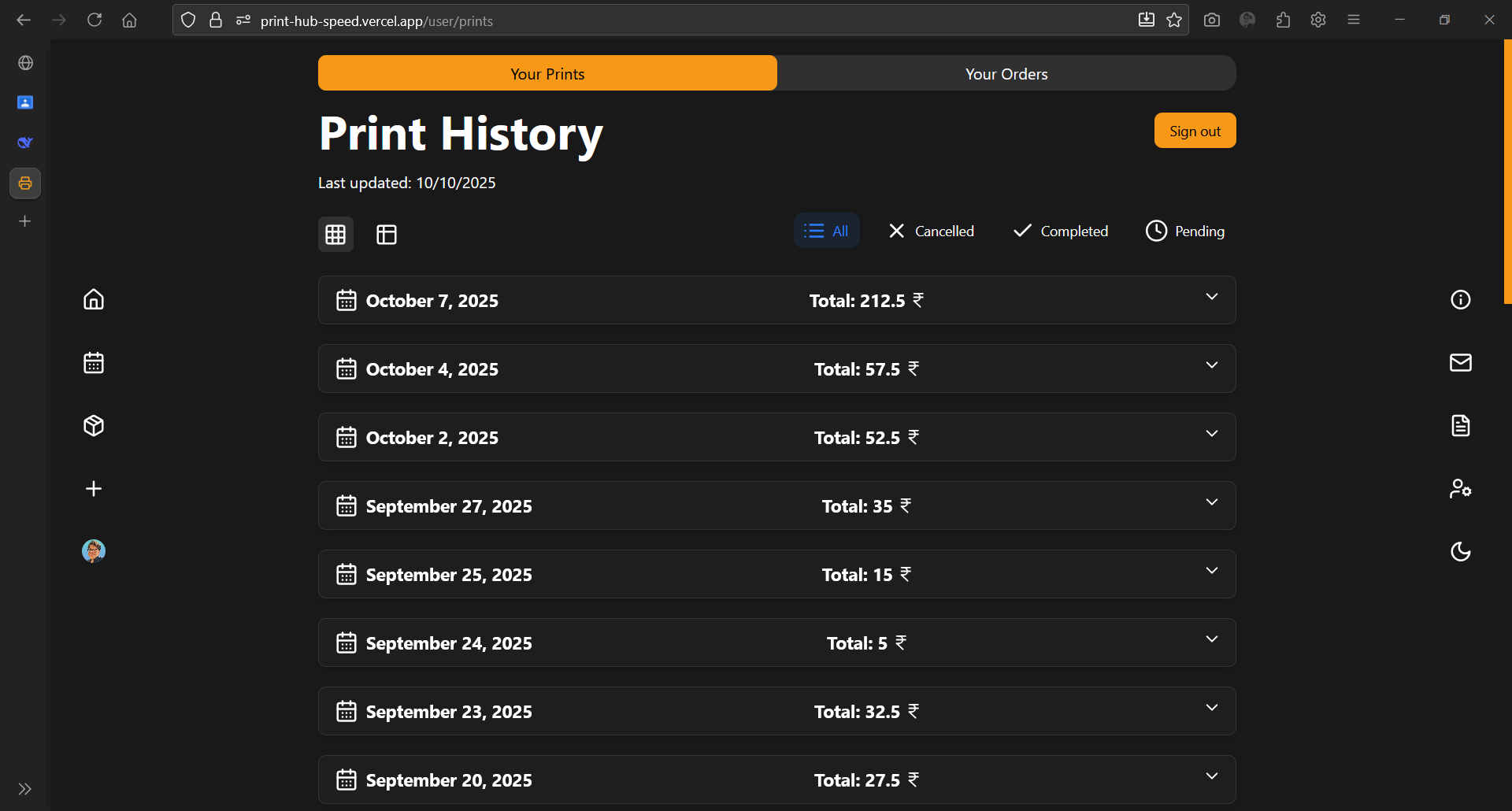
{ status: 500 }

);

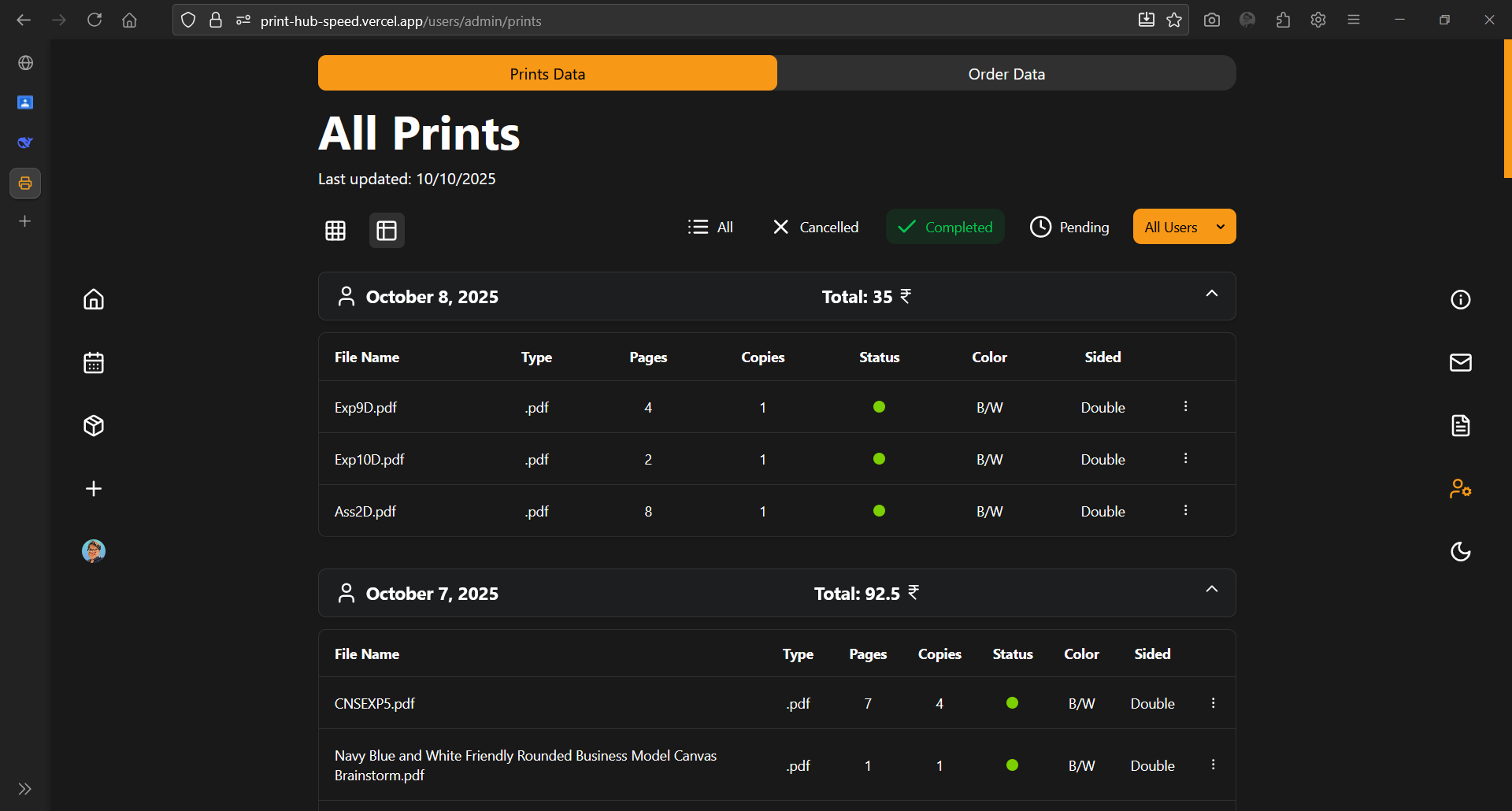
}

}

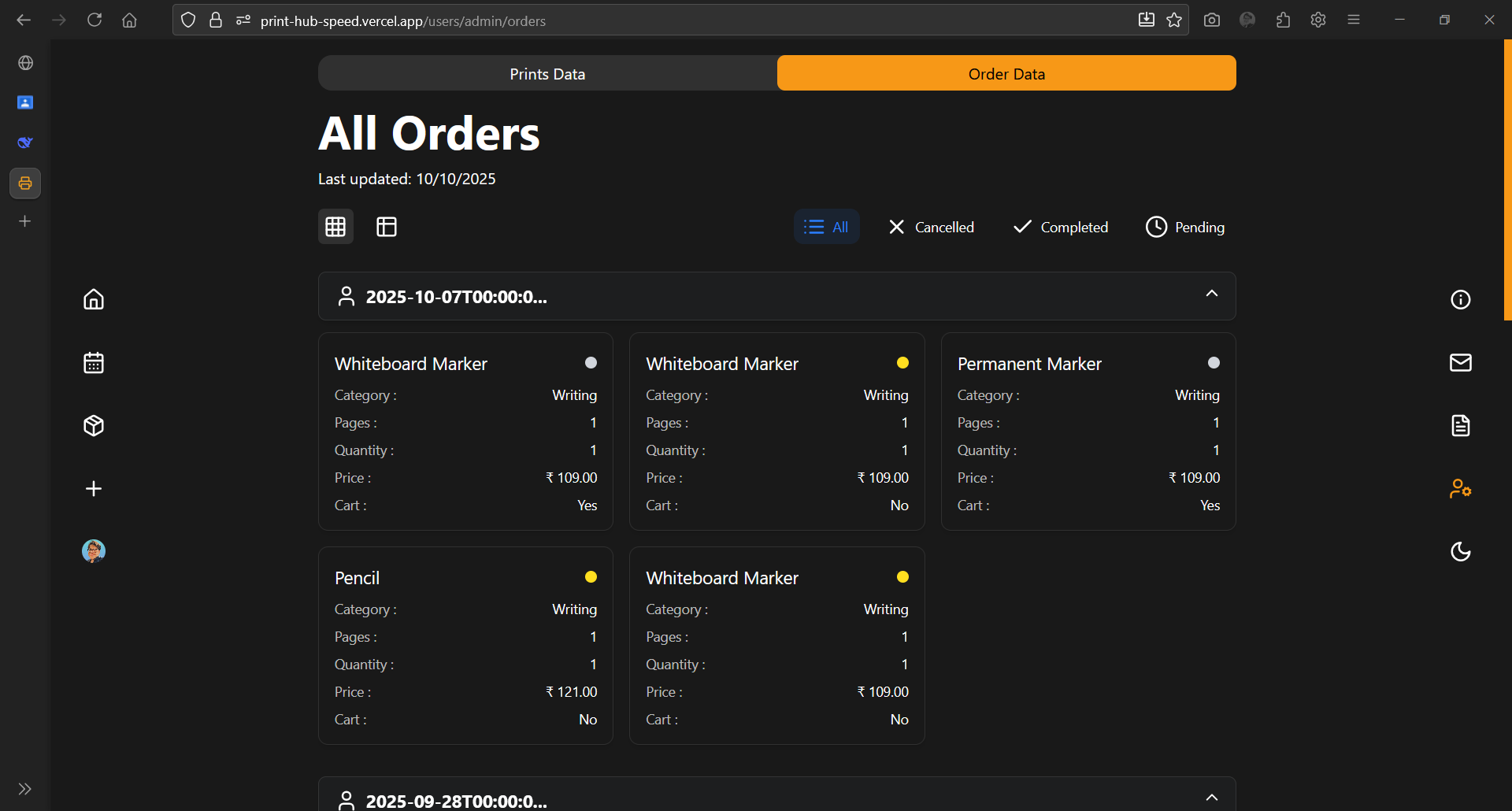
OUTPUT:



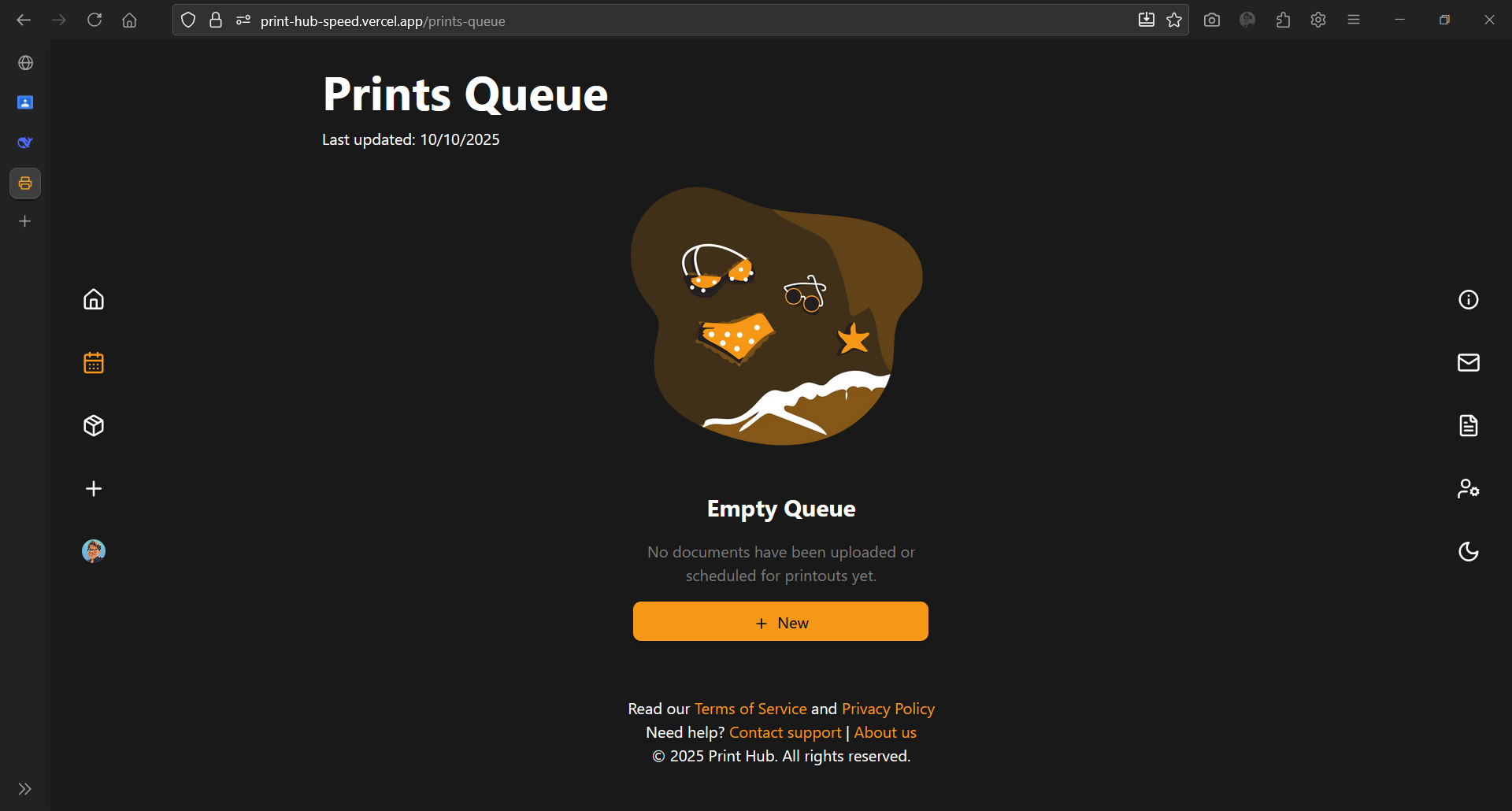
User History (Prints) Page



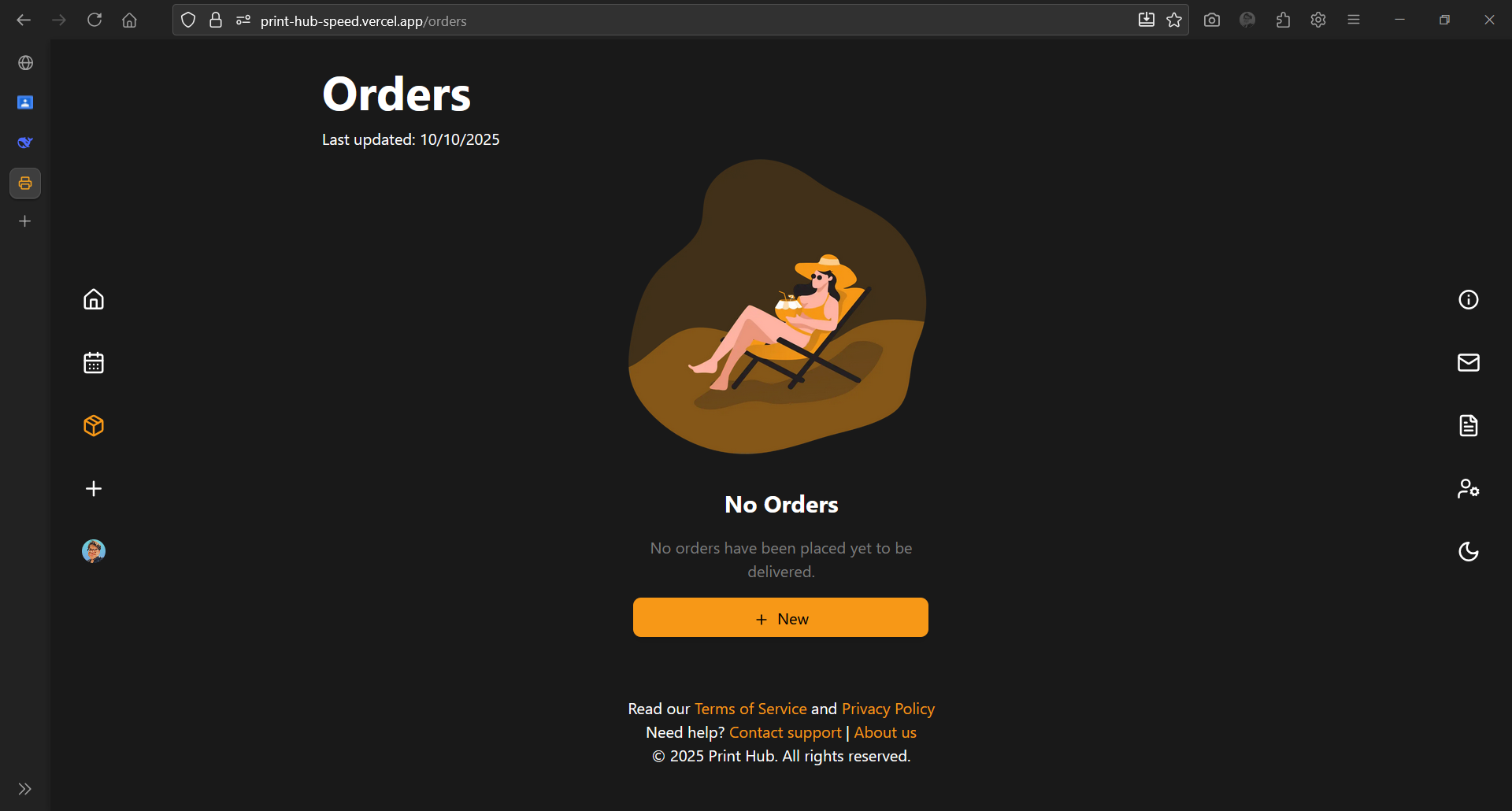
All orders Page (Admin)



All orders Page (Admin)



Prints Queue Page



Pending Orders Page