Name: Abdurrahman Qureshi

Roll No: 242466

Assignment No: 2

Date Of Performance: 10/10/2025

Q: Implementation of Nodejs/ Express Rest API applications.

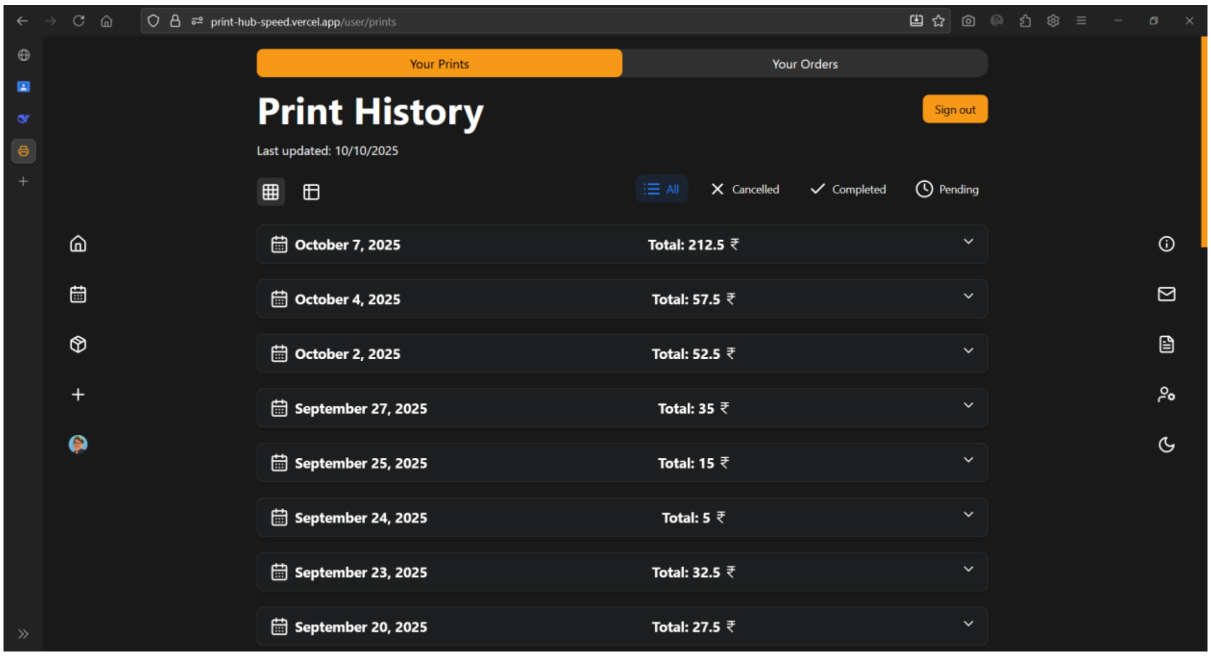
<u>CODE (src/app/api/*):</u>

```
==> 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 {
        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 = 'pending',
                in_cart = false
            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/prints/complete-print/route.ts
import { NextResponse } from "next/server";
import pool from "@/lib/neon/config";
export async function PATCH(req: Request) {
    try {
        const { print_id } = await req.json();
        if (!print_id) {
            return NextResponse.json(
                { error: "Print ID is required",
status: false },
                { status: 400 }
            );
        }
        const query = `
            UPDATE "prints"
            SET print_status = 'completed'
            WHERE print_id = $1
            RETURNING *;
        `;
        const values = [print_id];
        const result = await pool.query(query,
values);
        if (result.rows.length 0) {
            return NextResponse.json(
                { error: "Print not found",
status: false },
                { status: 404 }
            );
        }
        return NextResponse.json({
            data: result.rows[0],
            message: "Print cancelled
successfully",
            status: true,
        });
    } catch (error) {
        console.error("Error cancelling:",
error);
        return NextResponse.json(
            { error: String(error), status:
false },
            { status: 500 }
        );
    }
}

==> post/neon/orders/insert-record/route.ts
import { NextResponse } from "next/server";
import pool from "@/lib/neon/config";
import { getFormatDate } from
"@/functions/utility";
export async function POST(req: Request) {
    try {
        const body = await req.json();
        const { order, user } = body;
        const query = `
      INSERT INTO "orders" (
        item_id,
        user_id,
        user_name,
        item_name,
        item_category,
        item_type,
        item_quantity,
        item_price,
        instructions,
        ordered_at,
        order_status,
        in_cart
      ) VALUES ($1, $2, $3, $4, $5, $6, $7, $8,
$9, $10, $11, $12)
      RETURNING *;
    `;
        const values = [
            order.item_id,
            user.id,
            user.fullName,
            order.item_name,
            order.item_category,
            order.item_type,
            order.item_quantity,
            order.item_price,
            order.instructions,
            getFormatDate(new Date()),
            order.order_status,
            order.in_cart,
        ];
        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/orders/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 "orders"
      WHERE user_id = $1
      ORDER BY ordered_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/neon/prints/check-hash/route.ts
import { NextResponse } from "next/server";
import pool from "@/lib/neon/config";
export async function POST(req: Request) {
    try {
        const { hashed_content } = await
req.json();
        const query = `
      SELECT * FROM "prints"
      WHERE hashed_content = $1
    `;
        const result = await pool.query(query,
[hashed_content]);
        if (result.rows.length > 0) {
            return NextResponse.json({
                fileExists: true,
                status: true,
                numberOfRows:
result.rows.length,
                existsResult: {
                    hashed_content:
result.rows[0].hashed_content,
                    ipfs_id:
result.rows[0].ipfs_id,
                    ipfs_url:
result.rows[0].ipfs_link,
                },
            });
        }
        return NextResponse.json({
            fileExists: false,
            numberOfRows: result.rows.length,
            status: true,
        });
    } catch (error) {
        console.error("Error fetching user
history:", error);
        return NextResponse.json(
            { error: String(error), status:
false },
            { status: 500 }
        );
    }
}

==> post/neon/prints/insert-record/route.ts
import { NextResponse } from "next/server";
import pool from "@/lib/neon/config";
import { getFormatDate } from
"@/functions/utility";
import { generatePrintId } from
"@/functions/prints";
export async function POST(req: Request) {
    try {
        const body = await req.json();
        const { pinataResult, file, user } =
body;
        const query = `
      INSERT INTO "prints" (
        print_id,
        user_id,
        user_name,
        file_name,
        file_type,
        ipfs_id,
        ipfs_link,
        hashed_content,
        print_count,
        page_count,
        print_type,
        print_color,
        binding_type,
        instructions,
        print_status,
        uploaded_at
      ) VALUES ($1, $2, $3, $4, $5, $6, $7, $8,
$9, $10, $11, $12, $13, $14, $15, $16)
      RETURNING *;
    `;
        const values = [
            generatePrintId(),
            user.id,
            user.fullName,
            file.file_name,
            file.file_type,
            pinataResult.ipfs_id,
            pinataResult.ipfs_url,
            file.hashed_content,
            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:



print-hub-speed.vercel.app/user/prints

Your Prints | Your Orders

Print History

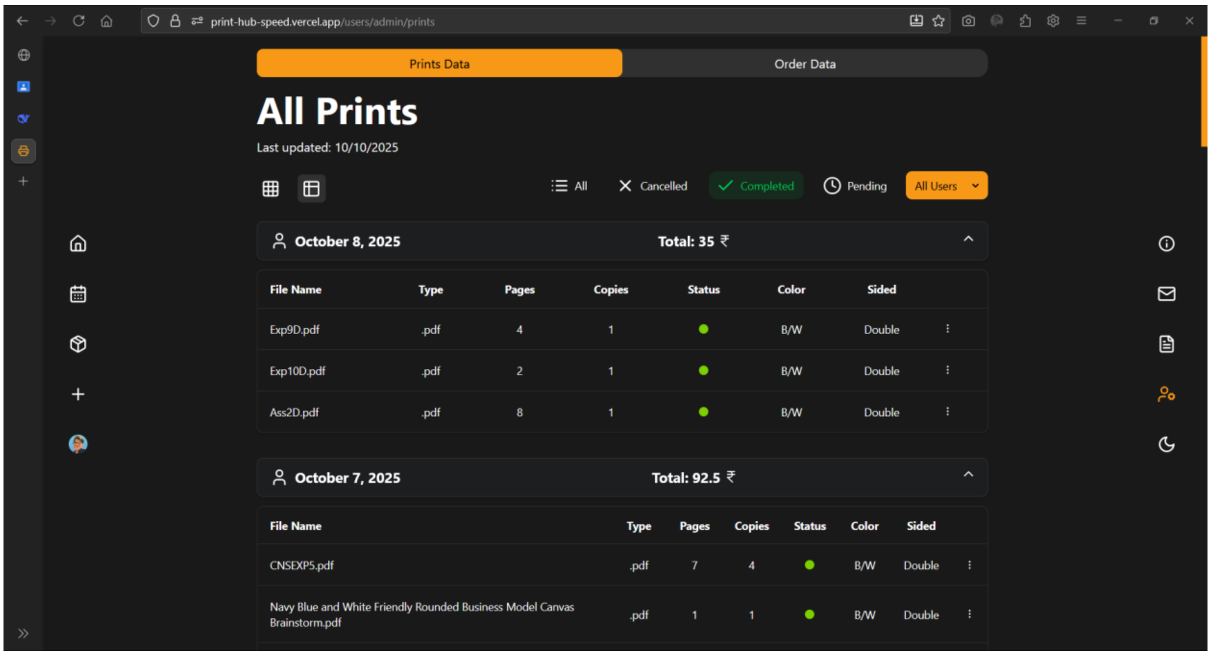
Sign out

Last updated: 10/10/2025

All | Cancelled | Completed | Pending

October 7, 2025 — Total: 212.5 ₹
October 4, 2025 — Total: 57.5 ₹
October 2, 2025 — Total: 52.5 ₹
September 27, 2025 — Total: 35 ₹
September 25, 2025 — Total: 15 ₹
September 24, 2025 — Total: 5 ₹
September 23, 2025 — Total: 32.5 ₹
September 20, 2025 — Total: 27.5 ₹

User History (Prints) Page



print-hub-speed.vercel.app/users/admin/prints

Prints Data | Order Data

All Prints

Last updated: 10/10/2025

All | Cancelled | Completed | Pending | All Users

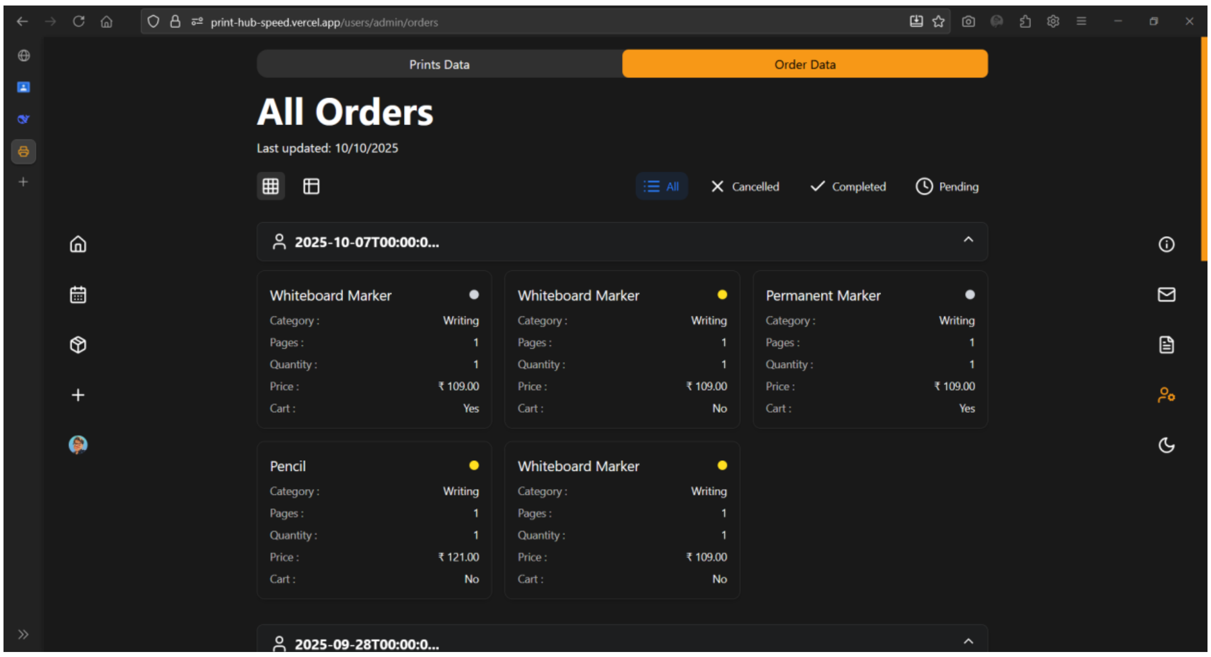
October 8, 2025 — Total: 35 ₹

File Name	Type	Pages	Copies	Status	Color	Sided
Exp9D.pdf	.pdf	4	1	●	B/W	Double
Exp10D.pdf	.pdf	2	1	●	B/W	Double
Ass2D.pdf	.pdf	8	1	●	B/W	Double

October 7, 2025 — Total: 92.5 ₹

File Name	Type	Pages	Copies	Status	Color	Sided
CNSEXP5.pdf	.pdf	7	4	●	B/W	Double
Navy Blue and White Friendly Rounded Business Model Canvas Brainstorm.pdf	.pdf	1	1	●	B/W	Double

All orders Page (Admin)



print-hub-speed.vercel.app/users/admin/orders

Prints Data | Order Data

All Orders

Last updated: 10/10/2025

All | Cancelled | Completed | Pending

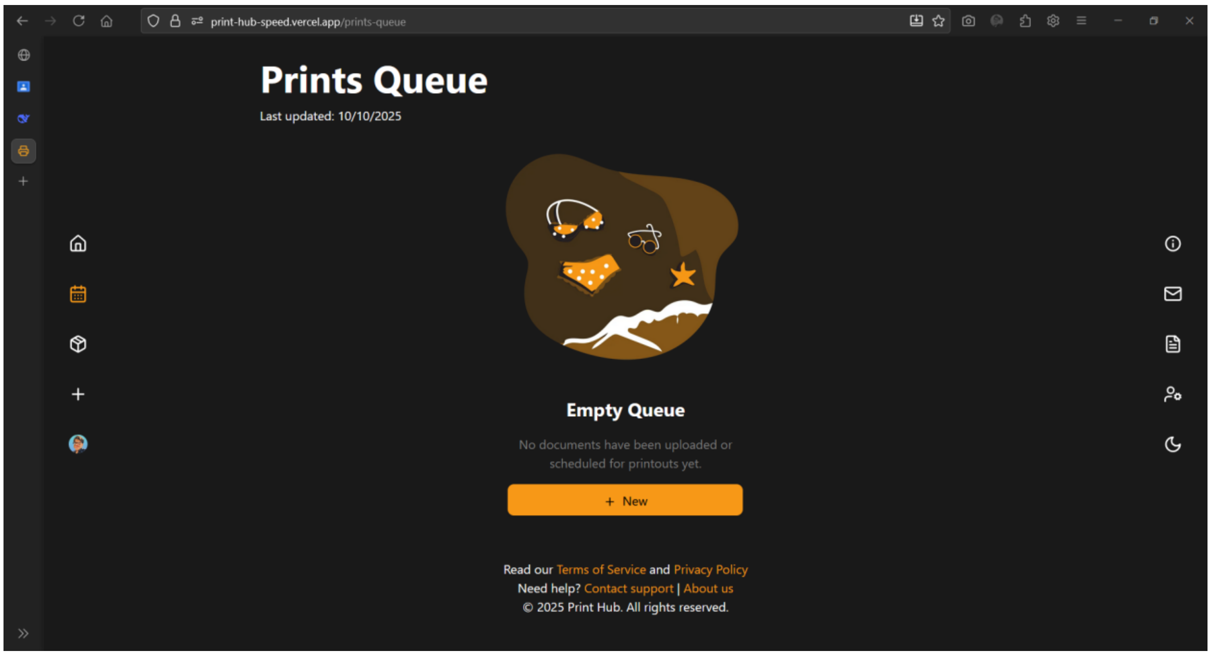
2025-10-07T00:00:0...

Whiteboard Marker	Whiteboard Marker	Permanent Marker
Category : Writing	Category : Writing	Category : Writing
Pages : 1	Pages : 1	Pages : 1
Quantity : 1	Quantity : 1	Quantity : 1
Price : ₹ 109.00	Price : ₹ 109.00	Price : ₹ 109.00
Cart : Yes	Cart : No	Cart : Yes

Pencil	Whiteboard Marker
Category : Writing	Category : Writing
Pages : 1	Pages : 1
Quantity : 1	Quantity : 1
Price : ₹ 121.00	Price : ₹ 109.00
Cart : No	Cart : No

2025-09-28T00:00:0...

All orders Page (Admin)


print-hub-speed.vercel.app/prints-queue

Prints Queue

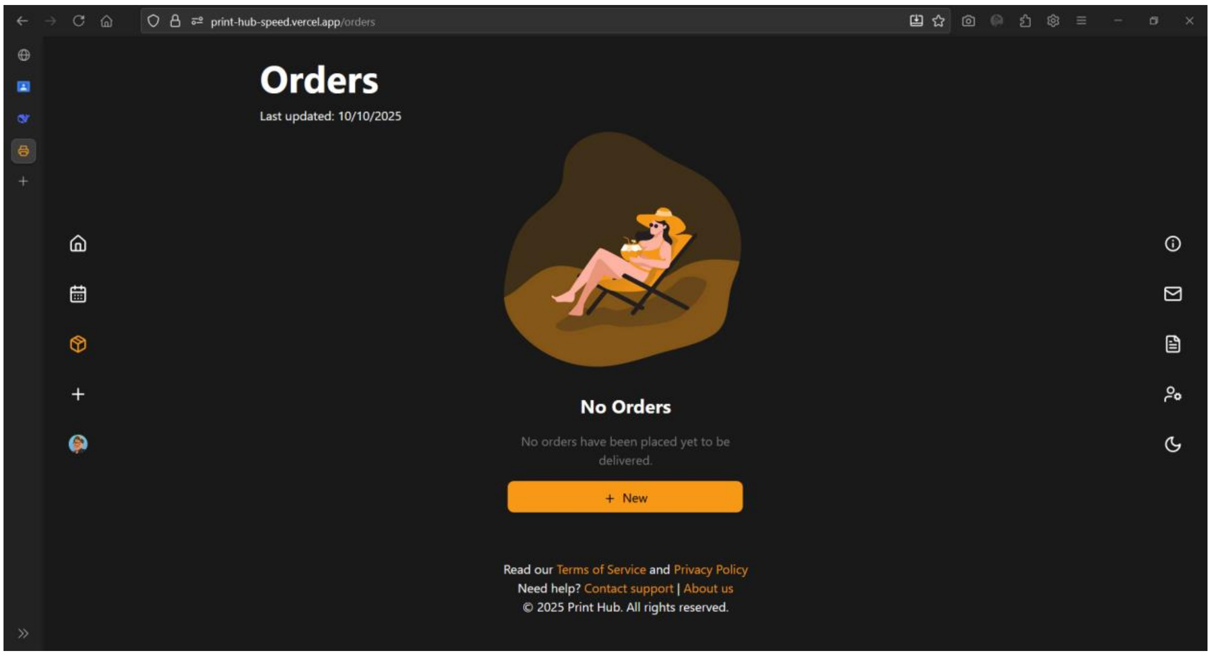
Last updated: 10/10/2025

Empty Queue

No documents have been uploaded or scheduled for printouts yet.

Read our Terms of Service and Privacy Policy
Need help? Contact support | About us
© 2025 Print Hub. All rights reserved.

Prints Queue Page


print-hub-speed.vercel.app/orders

Orders

Last updated: 10/10/2025

No Orders

No orders have been placed yet to be delivered.

Read our Terms of Service and Privacy Policy
Need help? Contact support | About us
© 2025 Print Hub. All rights reserved.

Pending Orders Page