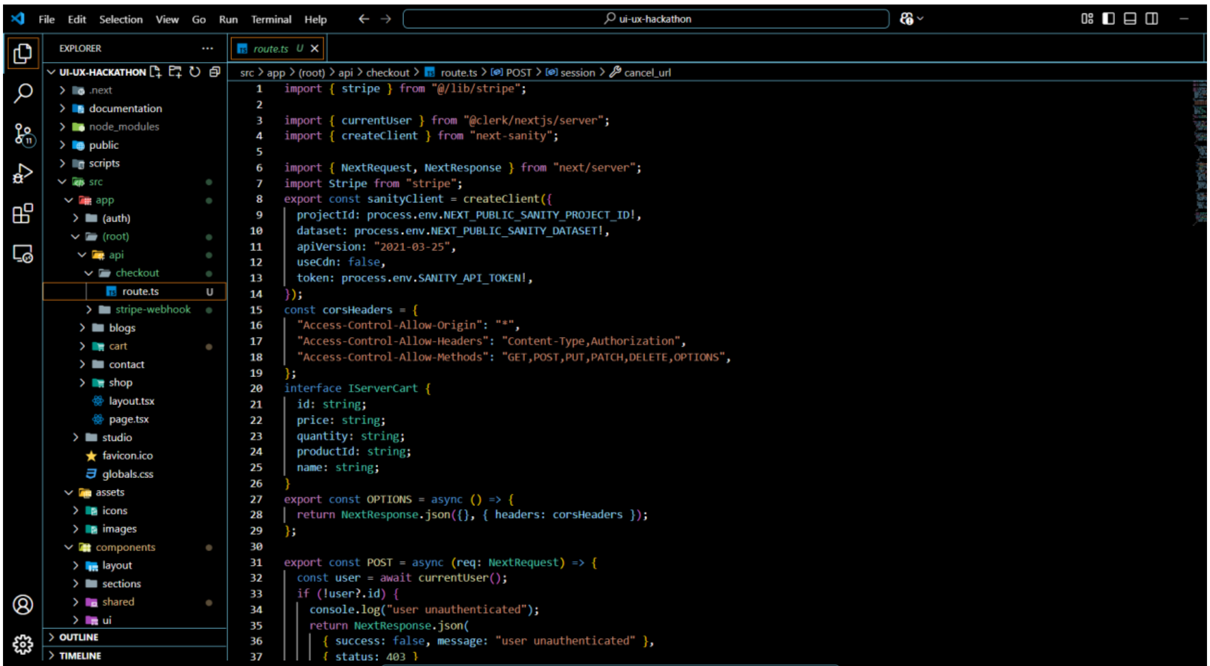
Day 6 – Deployment Preparation for – General - Ecommerce

Deployment Preparation and Staging Environment Setup

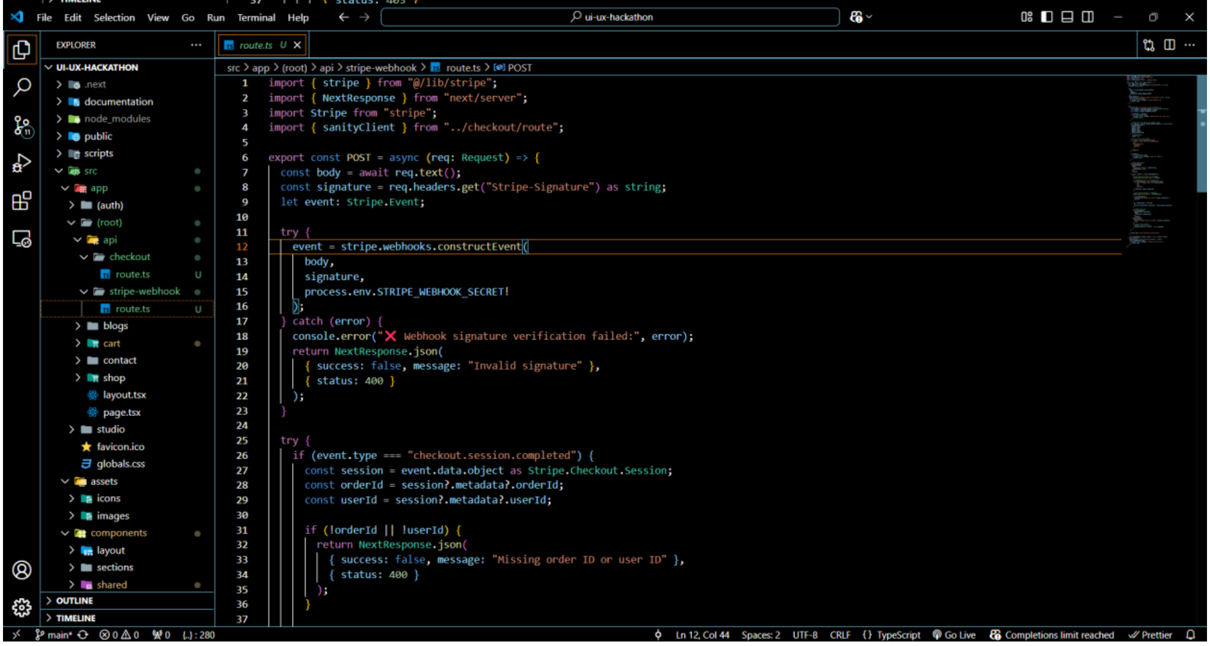
Objective

The primary objective of today is Deployment Preparation and Staging Environment Setup, focusing on ensuring seamless hosting, environment configuration, and staging deployment for thorough testing.

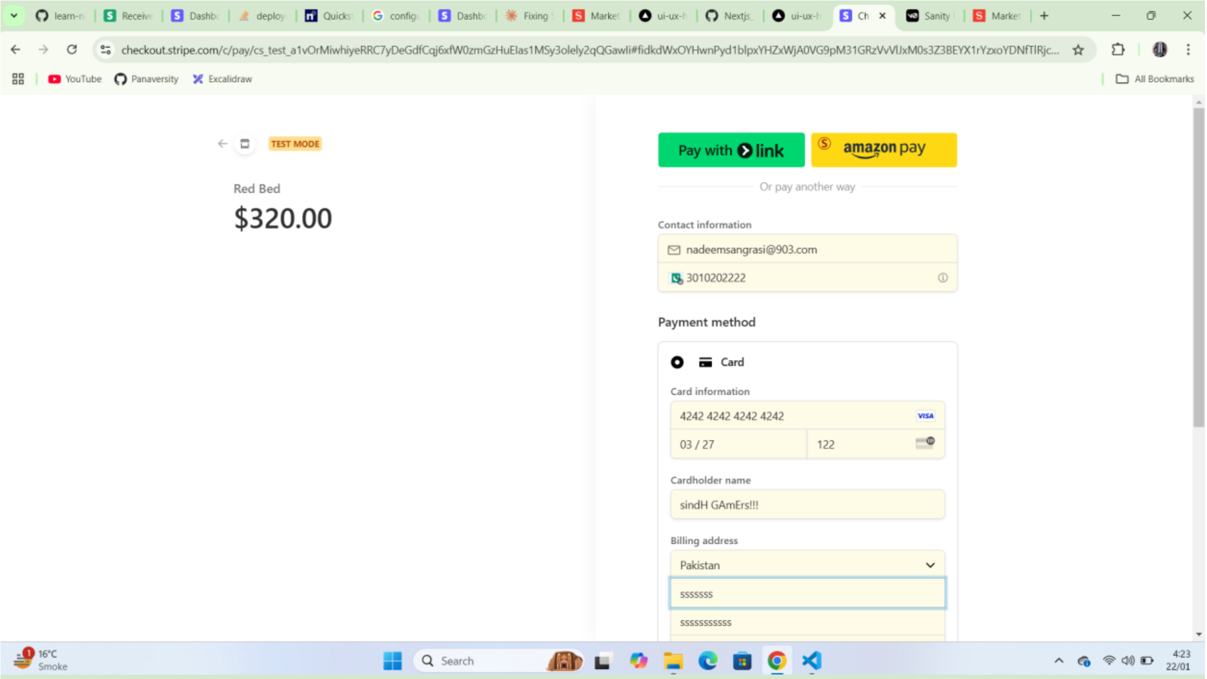
Additionally, I made a major change to my site by integrating the Stripe payment gateway using advanced techniques. Stripe Webhook was implemented to securely store order data into Sanity CMS, streamlining the order management process.



```
import { stripe } from "@/lib/stripe";

import { currentUser } from "@clerk/nextjs/server";
import { createClient } from "next-sanity";

import { NextRequest, NextResponse } from "next/server";
import Stripe from "stripe";
export const sanityClient = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID!,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET!,
  apiVersion: "2021-03-25",
  useCdn: false,
  token: process.env.SANITY_API_TOKEN!,
});
const corsHeaders = {
  "Access-Control-Allow-Origin": "*",
  "Access-Control-Allow-Headers": "Content-Type,Authorization",
  "Access-Control-Allow-Methods": "GET,POST,PUT,PATCH,DELETE,OPTIONS",
};
interface IServerCart {
  id: string;
  price: string;
  quantity: string;
  productId: string;
  name: string;
}
export const OPTIONS = async () => {
  return NextResponse.json({}, { headers: corsHeaders });
};

export const POST = async (req: NextRequest) => {
  const user = await currentUser();
  if (!user?.id) {
    console.log("user unauthenticated");
    return NextResponse.json(
      { success: false, message: "user unauthenticated" },
      { status: 403 }
```



```
import { stripe } from "@/lib/stripe";
import { NextResponse } from "next/server";
import Stripe from "stripe";
import { sanityClient } from "../checkout/route";

export const POST = async (req: Request) => {
  const body = await req.text();
  const signature = req.headers.get("Stripe-Signature") as string;
  let event: Stripe.Event;

  try {
    event = stripe.webhooks.constructEvent(
      body,
      signature,
      process.env.STRIPE_WEBHOOK_SECRET!
    );
  } catch (error) {
    console.error("❌ Webhook signature verification failed:", error);
    return NextResponse.json(
      { success: false, message: "Invalid signature" },
      { status: 400 }
    );
  }

  try {
    if (event.type === "checkout.session.completed") {
      const session = event.data.object as Stripe.Checkout.Session;
      const orderId = session?.metadata?.orderId;
      const userId = session?.metadata?.userId;

      if (!orderId || !userId) {
        return NextResponse.json(
          { success: false, message: "Missing order ID or user ID" },
          { status: 400 }
        );
      }

```



checkout.stripe.com/c/pay/cs_test_a1vOrMiwhiyeRRC7yDeGdfCqj6xfW0zmGzHuElas1MSy3olely2qQGawli#fidkdWxOYHwnPyd1blpxYHZxWjA0VG9pM31GRzVvVlJxM0s3Z3BEYX1rYzxoYDNfTlRjc...

Red Bed

$320.00

Contact information

nadeemsangrasi@903.com

3010202222

Payment method

● Card

Card information

4242 4242 4242 4242

03 / 27 122

Cardholder name

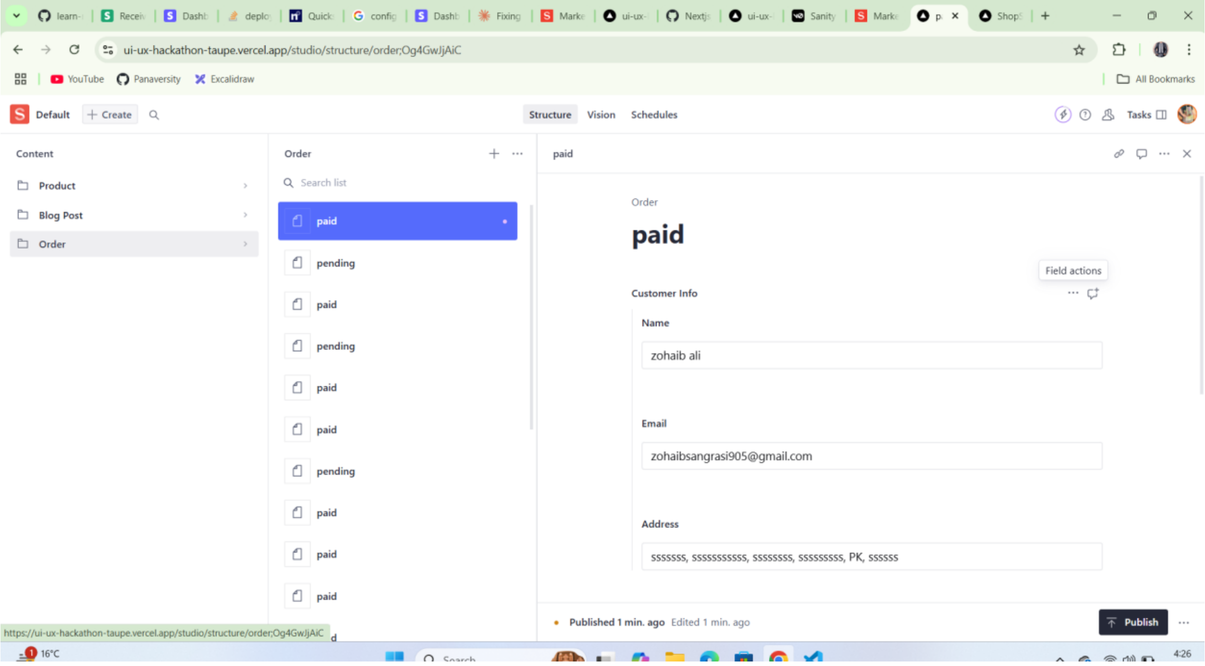
sindH GAmErs!!!

Billing address

Pakistan

sssssss

sssssssssss



ui-ux-hackathon-taupe.vercel.app/studio/structure/order;Og4GwJjAiC

Content

- Product
- Blog Post
- Order

Order

Search list

- paid
- pending
- paid
- pending
- paid
- paid
- pending
- paid
- paid
- paid

paid

Order

paid

Customer Info

Name

zohaib ali

Email

zohaibsangrasi905@gmail.com

Address

sssssss, sssssssssss, ssssssss, sssssssss, PK, ssssss

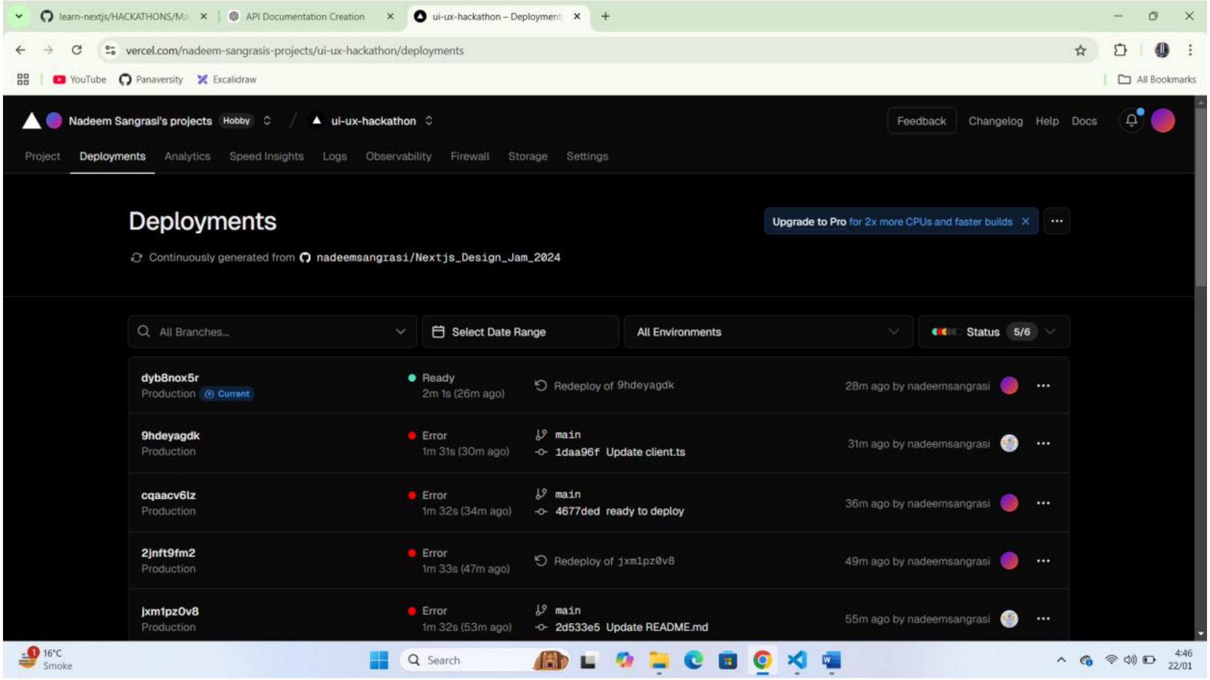
Published 1 min. ago Edited 1 min. ago

https://ui-ux-hackathon-taupe.vercel.app/studio/structure/order;Og4GwJjAiC

Today I Implemented These Steps

Step 1: Hosting Platform Setup

1. **Choose a Platform:**
   - Selected **Vercel** for deployment.
2. **Connect Repository:**
   - Linked the GitHub repository to Vercel.
   - Configured build settings and scripts for deployment.



Deployments

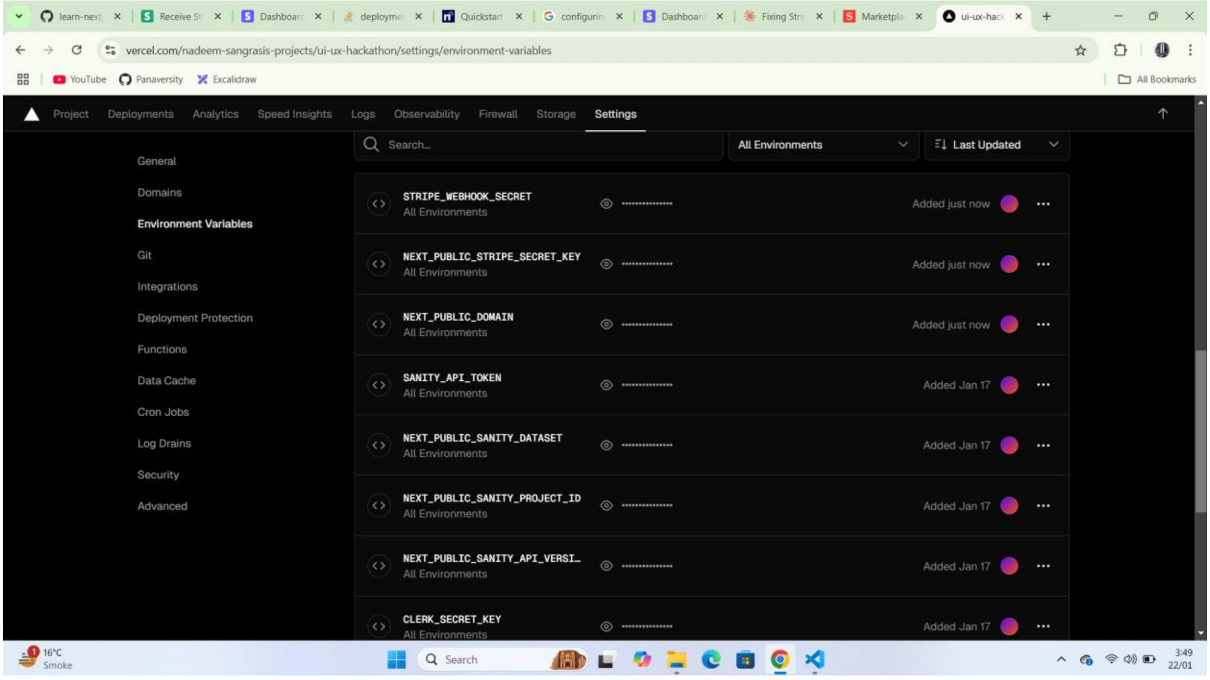
Continuously generated from nadeemsangrasi/Nextjs_Design_Jam_2024

dyb8nox5r Production Current	● Ready 2m 1s (26m ago)	Redeploy of 9hdeyagdk	28m ago by nadeemsangrasi
9hdeyagdk Production	● Error 1m 31s (30m ago)	main 1daa96f Update client.ts	31m ago by nadeemsangrasi
cqaacv6lz Production	● Error 1m 32s (34m ago)	main 4677ded ready to deploy	36m ago by nadeemsangrasi
2jnft9fm2 Production	● Error 1m 33s (47m ago)	Redeploy of jxm1pz0v8	49m ago by nadeemsangrasi
jxm1pz0v8 Production	● Error 1m 32s (53m ago)	main 2d533e5 Update README.md	55m ago by nadeemsangrasi

Step 2: Configure Environment Variables:

1. **Set Up Environment Variables:**
   - Added the following secrets:
     - **Clerk Secrets**
     - NEXT_PUBLIC_CLERK_PUBLISHABLE_KEY="<redacted>"
       CLERK_SECRET_KEY="<redacted>"
     - **Sanity Secrets**
     - NEXT_PUBLIC_SANITY_API_VERSION="2025-01-15"
     - NEXT_PUBLIC_SANITY_PROJECT_ID=""
     - NEXT_PUBLIC_SANITY_DATASET="production"
       SANITY_API_TOKEN="<redacted>"
     - **Stripe Keys**
     - NEXT_PUBLIC_STRIPE_SECRET_KEY="<redacted>"
       STRIPE_WEBHOOK_SECRET="<redacted>"
     - **Public Domain**

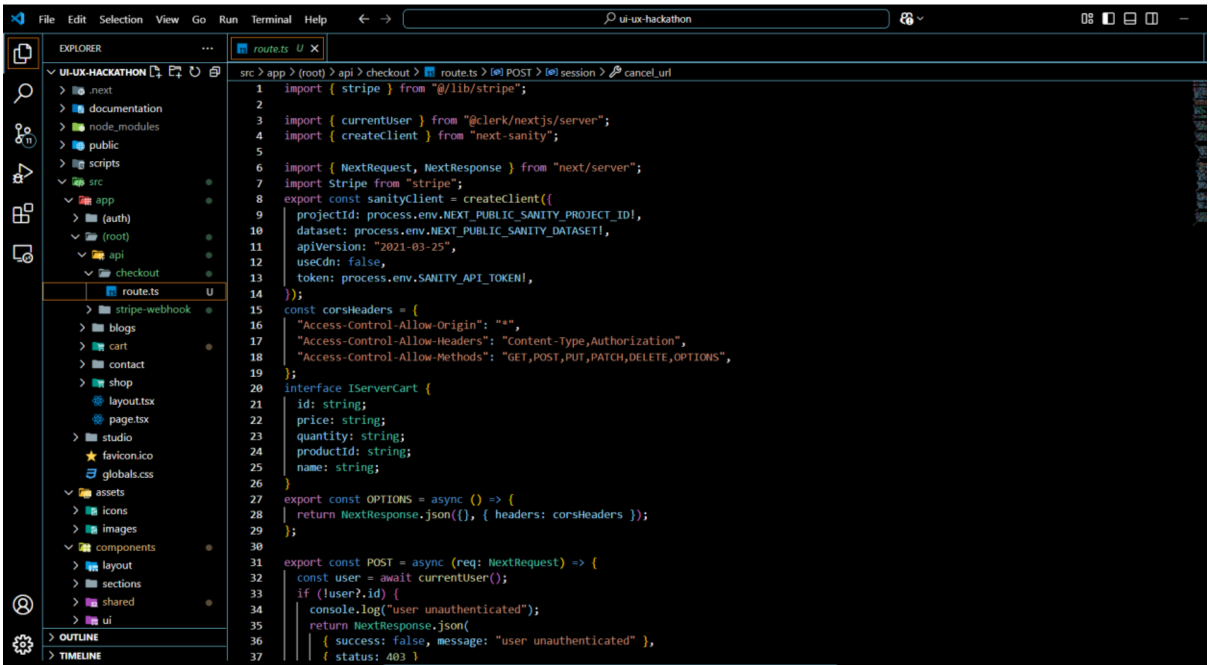
       NEXT_PUBLIC_DOMAIN="http://localhost:3000"



STRIPE_WEBHOOK_SECRET All Environments	•••••••••••••••	Added just now
NEXT_PUBLIC_STRIPE_SECRET_KEY All Environments	•••••••••••••••	Added just now
NEXT_PUBLIC_DOMAIN All Environments	•••••••••••••••	Added just now
SANITY_API_TOKEN All Environments	•••••••••••••••	Added Jan 17
NEXT_PUBLIC_SANITY_DATASET All Environments	•••••••••••••••	Added Jan 17
NEXT_PUBLIC_SANITY_PROJECT_ID All Environments	•••••••••••••••	Added Jan 17
NEXT_PUBLIC_SANITY_API_VERSI... All Environments	•••••••••••••••	Added Jan 17
CLERK_SECRET_KEY All Environments	•••••••••••••••	Added Jan 17

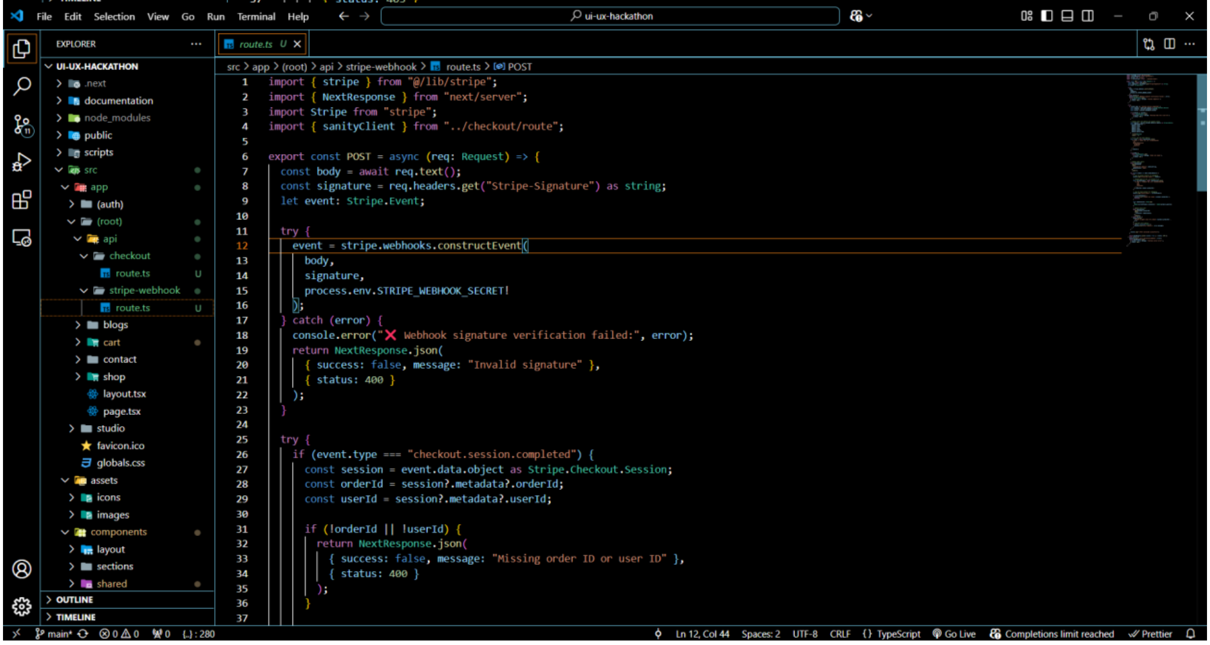
Step 3: Major Changes in Application

1. **Integration of Stripe Payment Gateway:**
   - Configured Stripe payment gateway for processing orders.
   - Implemented Stripe Webhooks to store order data in Sanity CMS.



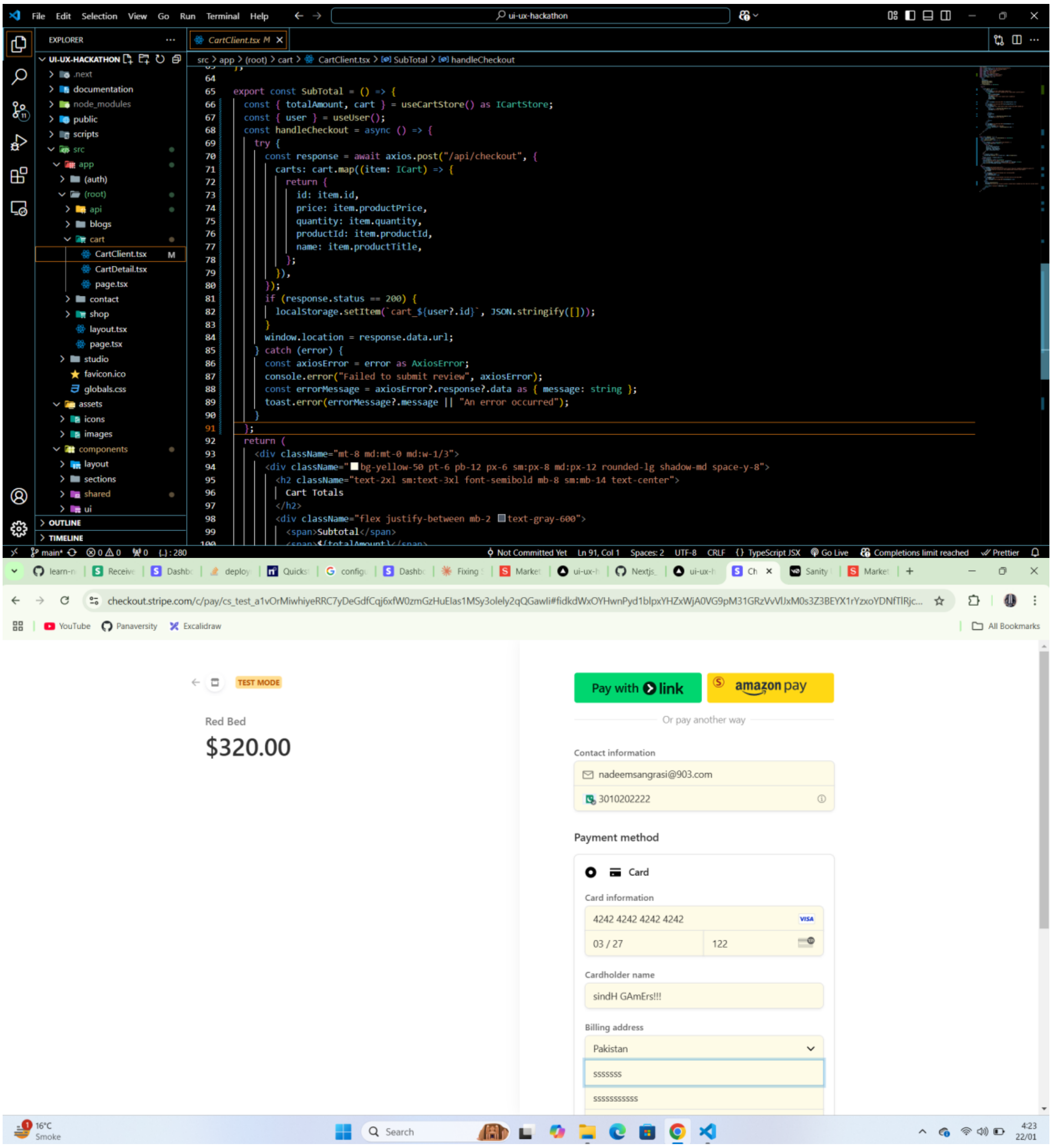
ui-ux-hackathon
route.ts
src > app > (root) > api > checkout > route.ts > POST > session > cancel_url

```
import { stripe } from "@/lib/stripe";

import { currentUser } from "@clerk/nextjs/server";
import { createClient } from "next-sanity";

import { NextRequest, NextResponse } from "next/server";
import Stripe from "stripe";
export const sanityClient = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID!,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET!,
  apiVersion: "2021-03-25",
  useCdn: false,
  token: process.env.SANITY_API_TOKEN!,
});
const corsHeaders = {
  "Access-Control-Allow-Origin": "*",
  "Access-Control-Allow-Headers": "Content-Type,Authorization",
  "Access-Control-Allow-Methods": "GET,POST,PUT,PATCH,DELETE,OPTIONS",
};
interface IServerCart {
  id: string;
  price: string;
  quantity: string;
  productId: string;
  name: string;
}
export const OPTIONS = async () => {
  return NextResponse.json({}, { headers: corsHeaders });
};

export const POST = async (req: NextRequest) => {
  const user = await currentUser();
  if (!user?.id) {
    console.log("user unauthenticated");
    return NextResponse.json(
      { success: false, message: "user unauthenticated" },
      { status: 403 }
```



ui-ux-hackathon
route.ts
src > app > (root) > api > stripe-webhook > route.ts > POST
Ln 12, Col 44
Spaces: 2
UTF-8
CRLF
TypeScript
Go Live
Completions limit reached
Prettier

```
import { stripe } from "@/lib/stripe";
import { NextResponse } from "next/server";
import Stripe from "stripe";
import { sanityClient } from "../checkout/route";

export const POST = async (req: Request) => {
  const body = await req.text();
  const signature = req.headers.get("Stripe-Signature") as string;
  let event: Stripe.Event;

  try {
    event = stripe.webhooks.constructEvent(
      body,
      signature,
      process.env.STRIPE_WEBHOOK_SECRET!
    );
  } catch (error) {
    console.error("❌ Webhook signature verification failed:", error);
    return NextResponse.json(
      { success: false, message: "Invalid signature" },
      { status: 400 }
    );
  }

  try {
    if (event.type === "checkout.session.completed") {
      const session = event.data.object as Stripe.Checkout.Session;
      const orderId = session?.metadata?.orderId;
      const userId = session?.metadata?.userId;

      if (!orderId || !userId) {
        return NextResponse.json(
          { success: false, message: "Missing order ID or user ID" },
          { status: 400 }
        );
      }
```



```
export const SubTotal = () => {
  const { totalAmount, cart } = useCartStore() as ICartStore;
  const { user } = useUser();
  const handleCheckout = async () => {
    try {
      const response = await axios.post("/api/checkout", {
        carts: cart.map((item: ICart) => {
          return {
            id: item.id,
            price: item.productPrice,
            quantity: item.quantity,
            productId: item.productId,
            name: item.productTitle,
          };
        }),
      });
      if (response.status == 200) {
        localStorage.setItem(`cart_${user?.id}`, JSON.stringify([]));
      }
      window.location = response.data.url;
    } catch (error) {
      const axiosError = error as AxiosError;
      console.error("Failed to submit review", axiosError);
      const errorMessage = axiosError?.response?.data as { message: string };
      toast.error(errorMessage?.message || "An error occurred");
    }
  };
  return (
    <div className="mt-8 md:mt-0 md:w-1/3">
      <div className="bg-yellow-50 pt-6 pb-12 px-6 sm:px-8 md:px-12 rounded-lg shadow-md space-y-8">
        <h2 className="text-2xl sm:text-3xl font-semibold mb-8 sm:mb-14 text-center">
          Cart Totals
        </h2>
        <div className="flex justify-between mb-2 text-gray-600">
          <span>Subtotal</span>
          <span>${totalAmount}</span>
```

main* 0 0 0 {.}: 280 Not Committed Yet Ln 91, Col 1 Spaces: 2 UTF-8 CRLF {} TypeScript JSX Go Live Completions limit reached Prettier
learn-n Receiv Dashb deploy Quicks config Dashb Fixing Market ui-ux-h Nextjs ui-ux-h Ch Sanity Market
YouTube Panaversity Excalidraw All Bookmarks

checkout.stripe.com/c/pay/cs_test_a1vOrMiwhiyeRRC7yDeGdfCqj6xfW0zmGzHuElas1MSy3olely2qQGawli#fidkdWxOYHwnPyd1blpxYHZxWjA0VG9pM31GRzVvVlJxM0s3Z3BEYX1rYzxoYDNfTlRjc...

TEST MODE

Red Bed

$320.00

Contact information

nadeemsangrasi@903.com

3010202222

Payment method

● Card

Card information

4242 4242 4242 4242 VISA

03 / 27 122

Cardholder name

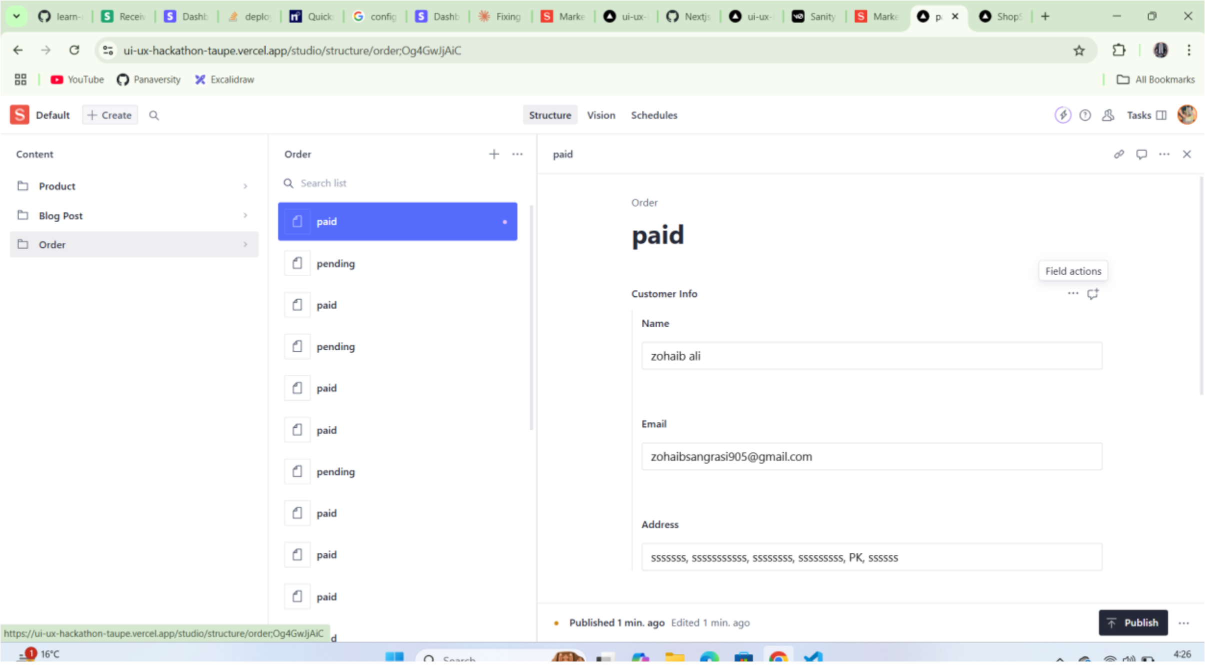
sindH GAmErs!!!

Billing address

Pakistan

sssssss

sssssssssss



Order

paid

Customer Info

Name: zohaib ali

Email: zohaibsangrasi905@gmail.com

Address: sssssss, sssssssssss, ssssssss, sssssssss, PK, ssssss

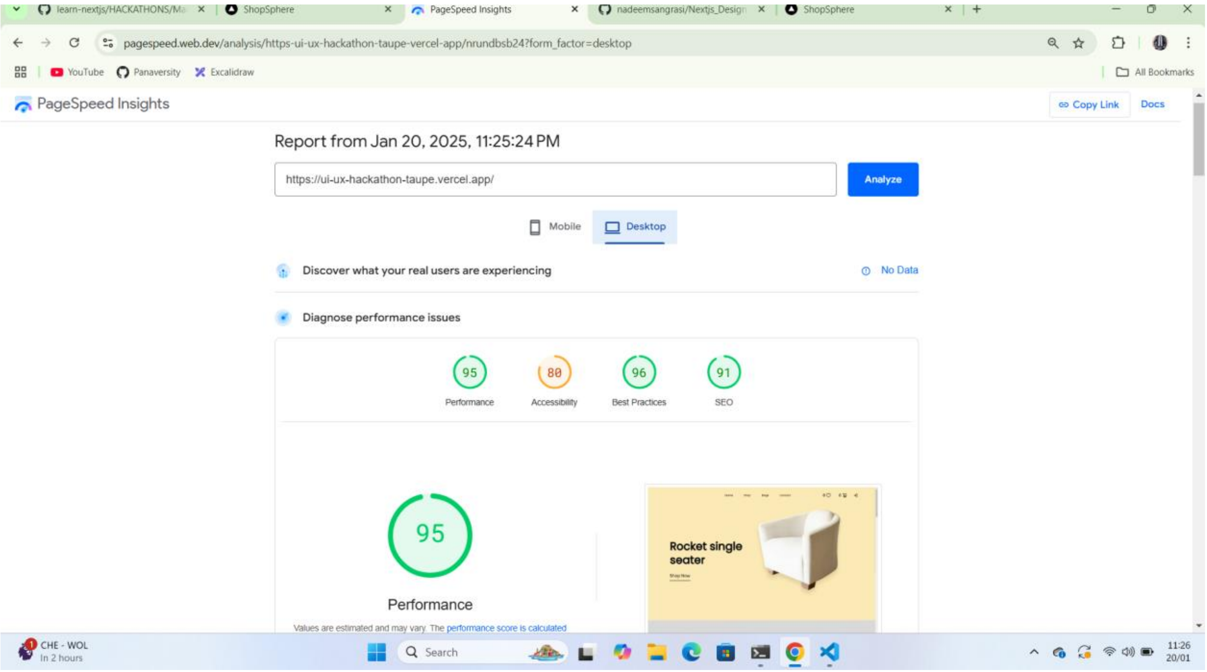
---

Step 4: Staging Deployment

1. **Deploy to Staging Environment:**
   - Deployed the application on Vercel's staging environment.
   - Ensured that the build process completed without errors.

Step 5: Staging Environment Testing

1. **Testing Types:**
   - Verified the following functionalities:
     - **Functional Testing:** Checked product listing, cart operations, and payment process.
     - **Performance Testing:** Used tools like Lighthouse for analyzing speed and performance.
     - **Cross-Browser Compatibility:** Validated the application across Chrome, Firefox, and Safari.



Report from Jan 20, 2025, 11:25:24 PM

https://ui-ux-hackathon-taupe.vercel.app/ Analyze

Mobile Desktop

Discover what your real users are experiencing — No Data

Diagnose performance issues

Performance	Accessibility	Best Practices	SEO
95	80	96	91

95

Performance

Values are estimated and may vary. The performance score is calculated

---

Step 6: Schema Changes

1. Updated Product Schema:

```
export const productSchema = {
  name: "product",
  title: "Product",
  type: "document",
  fields: [
    { name: "id", title: "ID", type: "string" },
    { name: "name", title: "Name", type: "string" },
    { name: "imagePath", title: "Image Path", type: "url" },
    { name: "price", title: "Price", type: "number" },
    { name: "description", title: "Description", type: "text" },
    { name: "discountPercentage", title: "Discount Percentage", type: "number" },
    { name: "isFeaturedProduct", title: "Is Featured Product", type: "boolean" },
    { name: "stockLevel", title: "Stock Level", type: "number" },
    { name: "category", title: "Category", type: "string" },
    {
      name: "images",
      title: "Images",
      type: "array",
      of: [
        {
          type: "image",
          options: { hotspot: true },
        },
      ],
    },
  ],
};
```

28. Updated Blog Schema:

```
export const blogSchema = {
  name: "blogPost",
  title: "Blog Post",
  type: "document",
  fields: [
    { name: "title", title: "Title", type: "string" },
    { name: "author", title: "Author", type: "string" },
    { name: "publishedAt", title: "Published At", type: "datetime" },
    { name: "category", title: "Category", type: "string" },
    { name: "excerpt", title: "Excerpt", type: "text", description: "A short summary of the post." },
    { name: "image", title: "Image", type: "image" },
  ],
};
```

Updated Order Schema:

```
export const orderSchema = {
  name: "order",
  title: "Order",
  type: "document",
  fields: [
    { name: "orderId", title: "Order ID", type: "string" },
    { name: "user", title: "User", type: "reference", to: [{ type: "user" }] },
    { name: "items", title: "Items", type: "array", of: [{ type: "reference", to: [{ type: "product" }] }] },
    { name: "totalAmount", title: "Total Amount", type: "number" },
    { name: "paymentStatus", title: "Payment Status", type: "string" },
    { name: "createdAt", title: "Created At", type: "datetime" },
  ],
};
```

Conclusion

Day 6 focused on critical updates and deployments, ensuring the application was ready for staging with enhanced functionalities, particularly the integration of Stripe and improved schema design.