BSc. (Hons) in Software Engineering
Faculty of Computing

Online Bookshop Assignment

Module: Advanced Web Technologies

Name: H.A.C.D.P. Hettiarachchi

Student Number: M20000210007

Table of Contents

1. User Interface Documentation Requirements 3

1.1. User Authentication Screens 3

1.2. Book Listing and Search 11

1.3. Shopping Cart 18

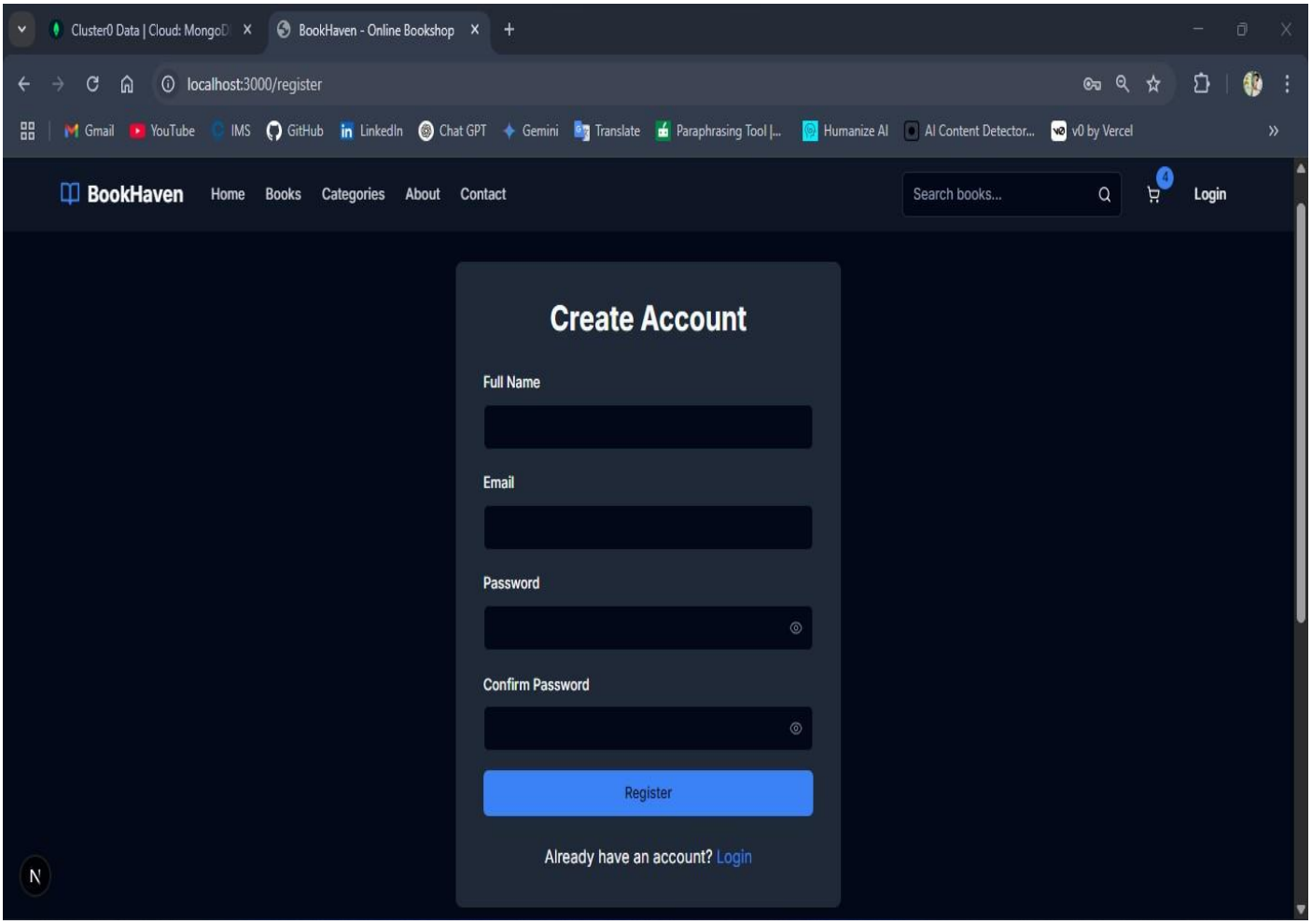
1.4. Checkout Process 23

1.5. User Profile 31

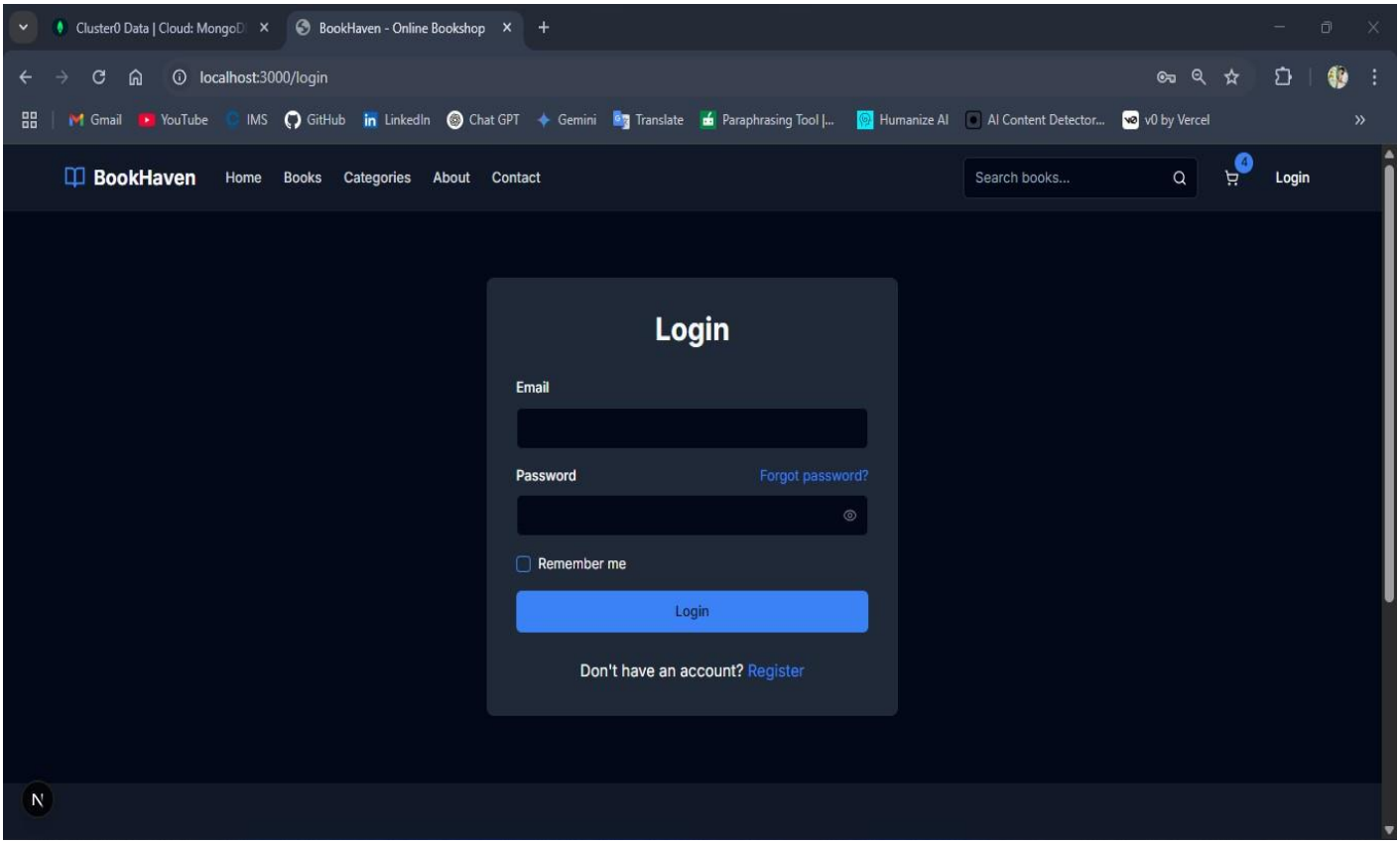
1. User Interface Documentation Requirements

1.1. User Authentication Screens

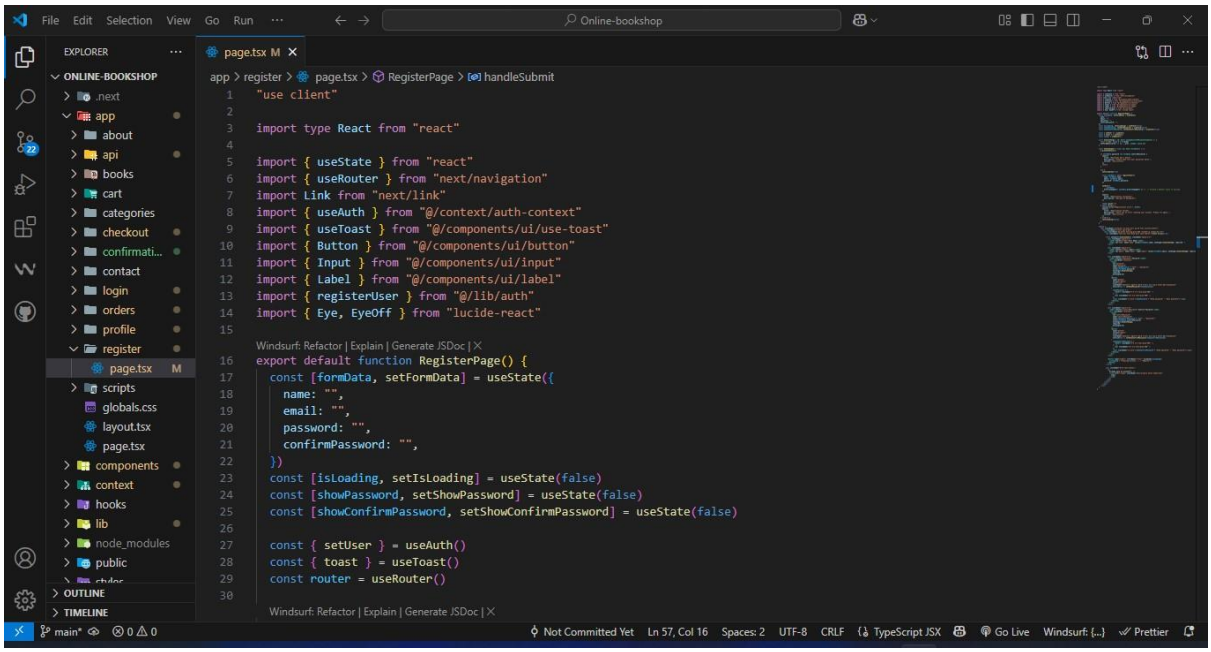
- Registration page screenshot

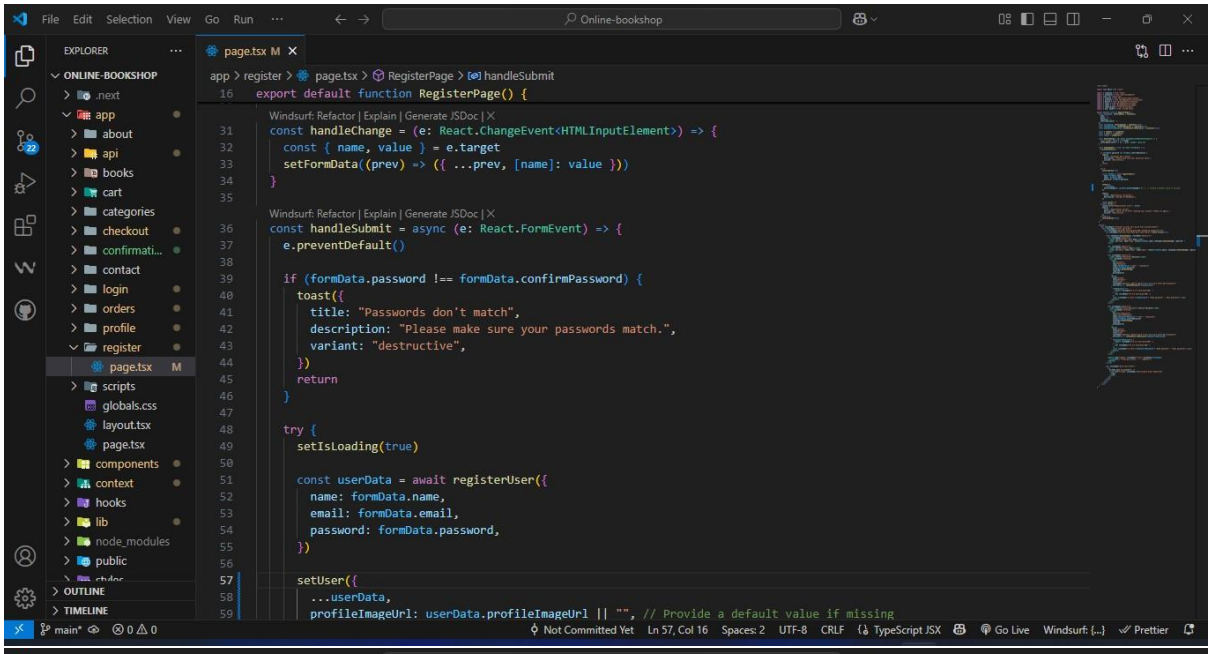


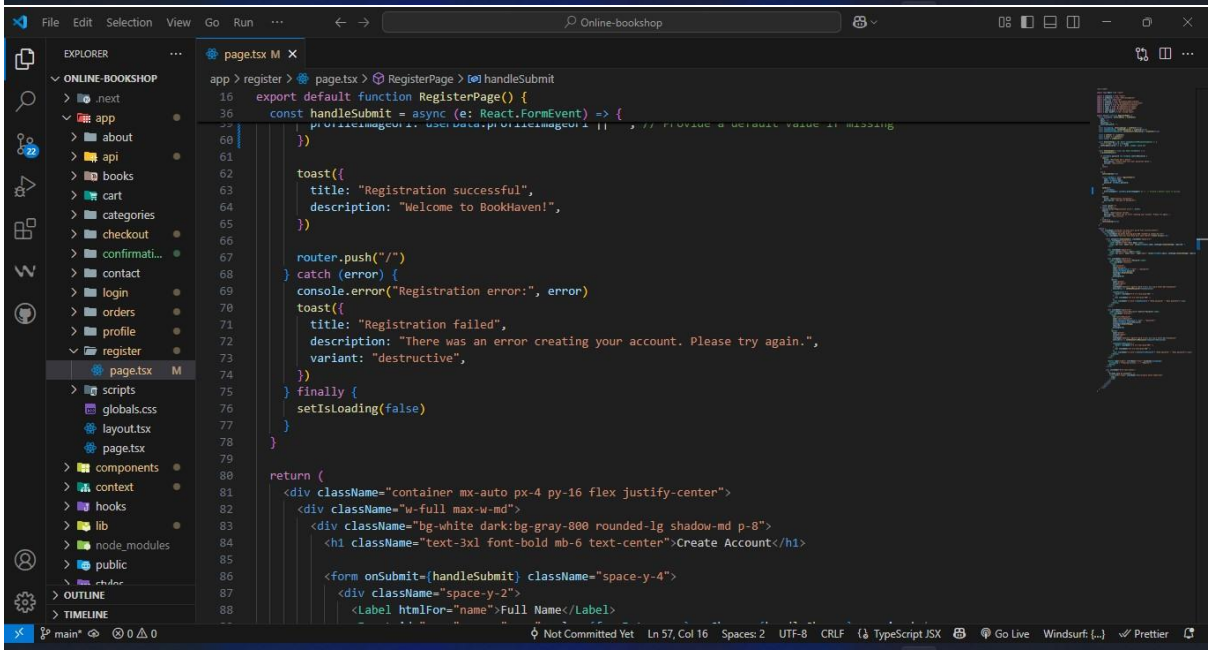
- Login page screenshot



- Associated HTML/CSS/JavaScript code screenshots for each page
  1.Registration page code screenshots

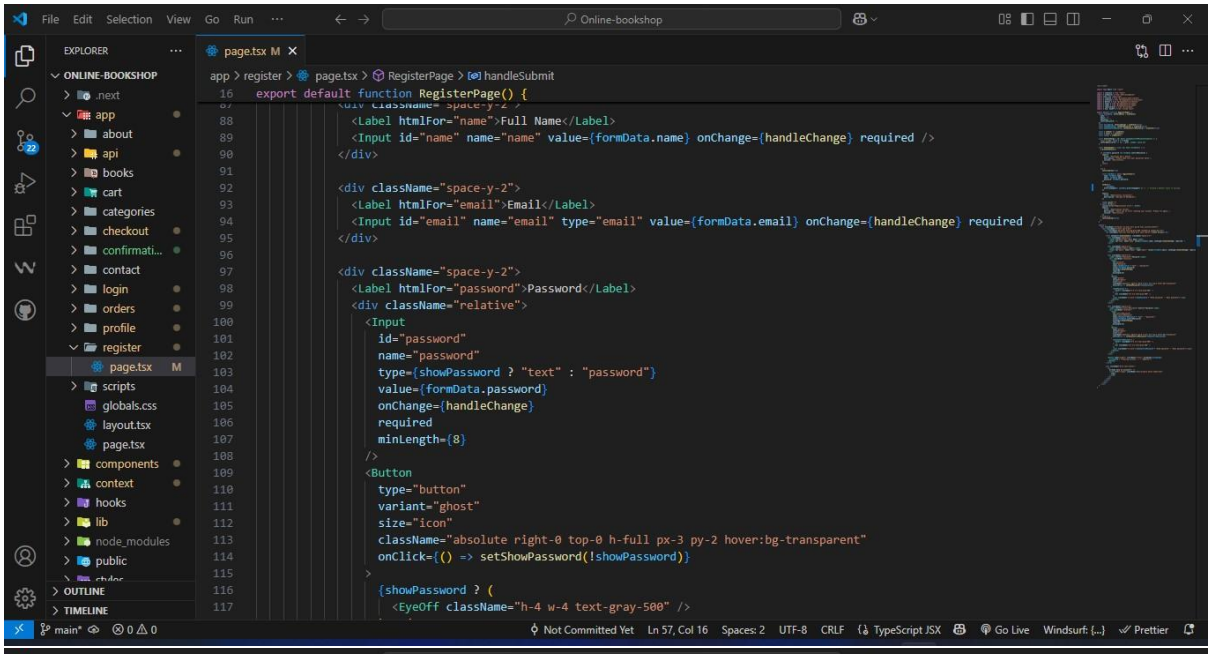


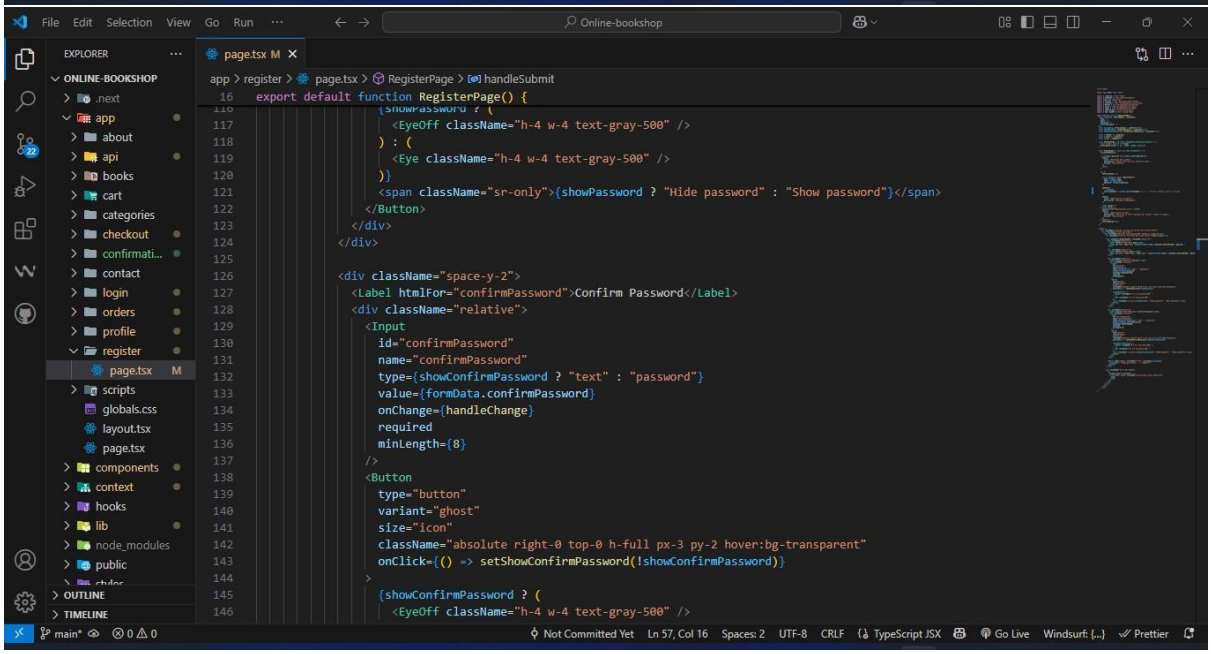

File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > register > page.tsx > RegisterPage > handleSubmit
ONLINE-BOOKSHOP
.next
app
about
api
books
cart
categories
checkout
confirmati...
contact
login
orders
profile
register
page.tsx M
scripts
globals.css
layout.tsx
page.tsx
components
context
hooks
lib
node_modules
public
OUTLINE
TIMELINE
export default function RegisterPage() {
Windsurf: Refactor | Explain | Generate JSDoc |
const handleChange = (e: React.ChangeEvent<HTMLInputElement>) => {
const { name, value } = e.target
setFormData((prev) => ({ ...prev, [name]: value }))
}
Windsurf: Refactor | Explain | Generate JSDoc |
const handleSubmit = async (e: React.FormEvent) => {
e.preventDefault()
if (formData.password !== formData.confirmPassword) {
toast({
title: "Passwords don't match",
description: "Please make sure your passwords match.",
variant: "destructive",
})
return
}
try {
setIsLoading(true)
const userData = await registerUser({
name: formData.name,
email: formData.email,
password: formData.password,
})
setUser({
...userData,
profileImageUrl: userData.profileImageUrl || "", // Provide a default value if missing
main*
Not Committed Yet
Ln 57, Col 16
Spaces: 2
UTF-8
CRLF
TypeScript JSX
Go Live
Windsurf: {...}
Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > register > page.tsx > RegisterPage > handleSubmit
ONLINE-BOOKSHOP
export default function RegisterPage() {
const handleSubmit = async (e: React.FormEvent) => {
})
toast({
title: "Registration successful",
description: "Welcome to BookHaven!",
})
router.push("/")
} catch (error) {
console.error("Registration error:", error)
toast({
title: "Registration failed",
description: "There was an error creating your account. Please try again.",
variant: "destructive",
})
} finally {
setIsLoading(false)
}
}
return (

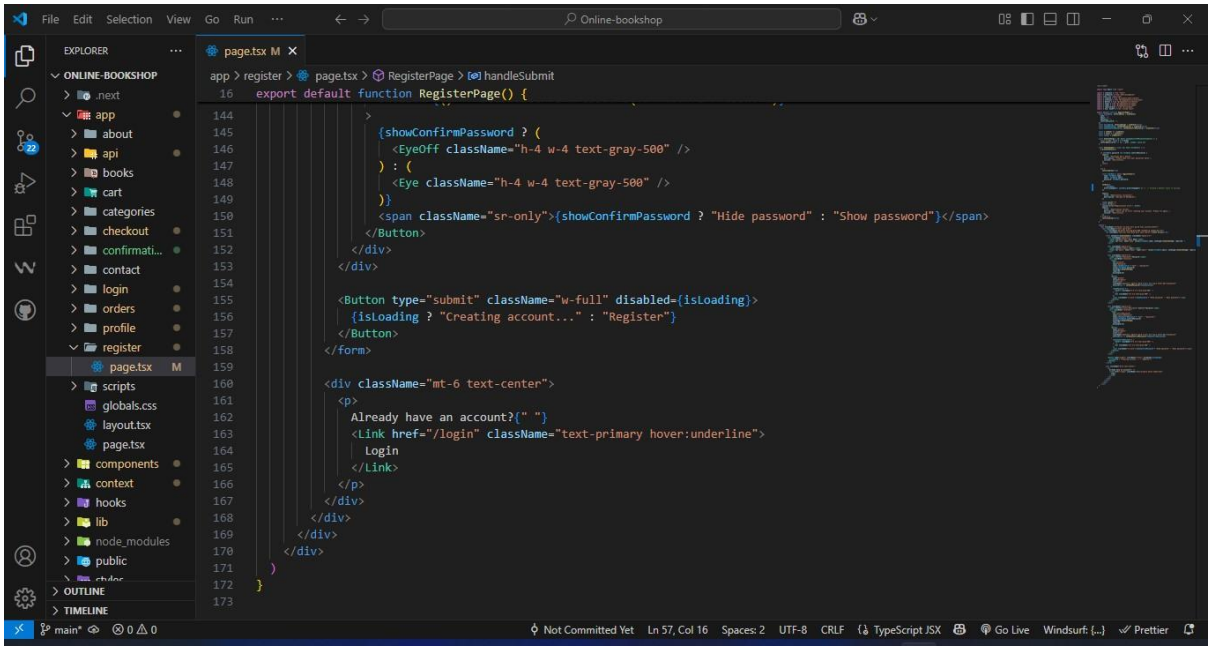
<h1 className="text-3xl font-bold mb-6 text-center">Create Account</h1>
<form onSubmit={handleSubmit} className="space-y-4">

<Label htmlFor="name">Full Name</Label>
main*
Not Committed Yet
Ln 57, Col 16
Spaces: 2
UTF-8
CRLF
TypeScript JSX
Go Live
Windsurf: {...}
Prettier

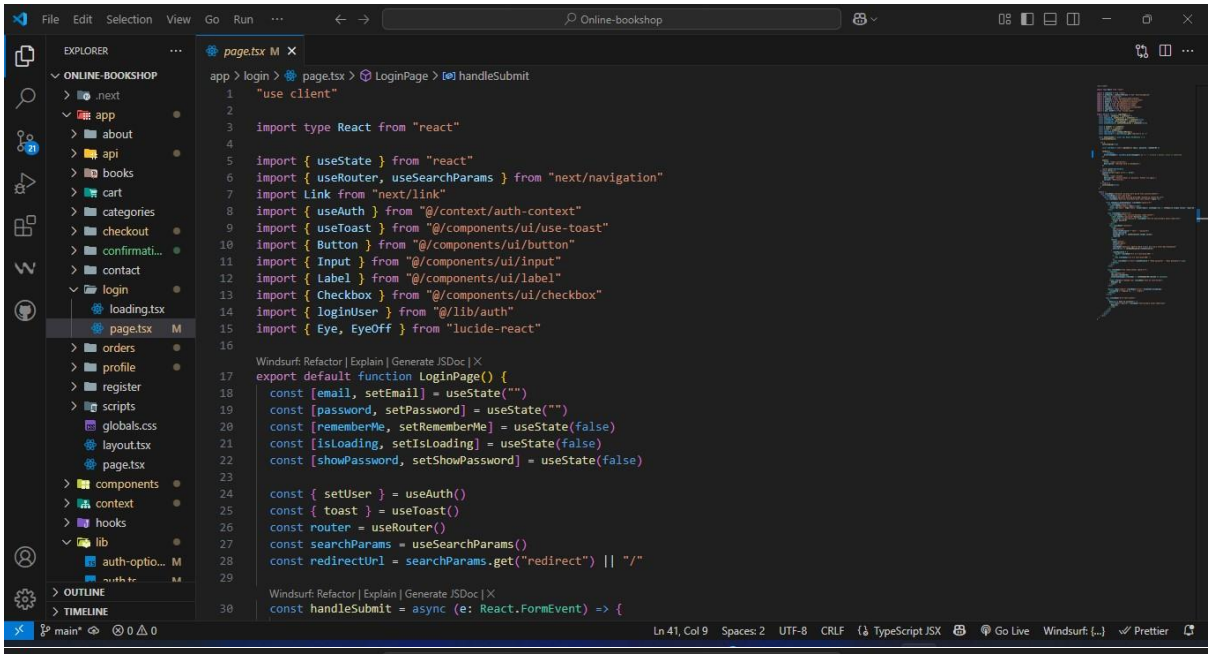

File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > register > page.tsx > RegisterPage > handleSubmit
ONLINE-BOOKSHOP
export default function RegisterPage() {
<Label htmlFor="name">Full Name</Label>
<Input id="name" name="name" value={formData.name} onChange={handleChange} required />
<Label htmlFor="email">Email</Label>
<Input id="email" name="email" type="email" value={formData.email} onChange={handleChange} required />
<Label htmlFor="password">Password</Label>
<Input
id="password"
name="password"
type={showPassword ? "text" : "password"}
value={formData.password}
onChange={handleChange}
required
minLength={8}
/>
<Button
type="button"
variant="ghost"
size="icon"
className="absolute right-0 top-0 h-full px-3 py-2 hover:bg-transparent"
onClick={() => setShowPassword(!showPassword)}
>
{showPassword ? (
<EyeOff className="h-4 w-4 text-gray-500" />
OUTLINE
TIMELINE
Not Committed Yet Ln 57, Col 16 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > register > page.tsx > RegisterPage > handleSubmit
ONLINE-BOOKSHOP
export default function RegisterPage() {
<EyeOff className="h-4 w-4 text-gray-500" />
) : (
<Eye className="h-4 w-4 text-gray-500" />
)}
{showPassword ? "Hide password" : "Show password"}
</Button>

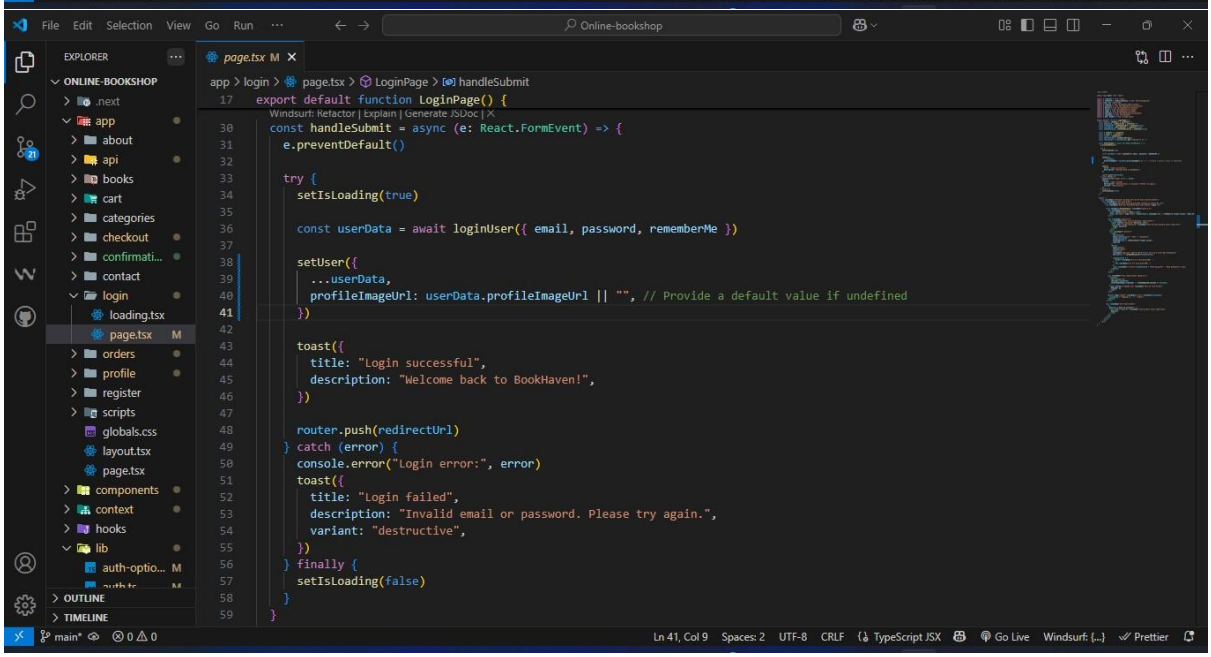
<Label htmlFor="confirmPassword">Confirm Password</Label>
<Input
id="confirmPassword"
name="confirmPassword"
type={showConfirmPassword ? "text" : "password"}
value={formData.confirmPassword}
onChange={handleChange}
required
minLength={8}
/>
<Button
type="button"
variant="ghost"
size="icon"
className="absolute right-0 top-0 h-full px-3 py-2 hover:bg-transparent"
onClick={() => setShowConfirmPassword(!showConfirmPassword)}
>
{showConfirmPassword ? (
<EyeOff className="h-4 w-4 text-gray-500" />
OUTLINE
TIMELINE
Not Committed Yet Ln 57, Col 16 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > register > page.tsx > RegisterPage > handleSubmit
ONLINE-BOOKSHOP
.next
app
about
api
books
cart
categories
checkout
confirmati...
contact
login
orders
profile
register
page.tsx M
scripts
globals.css
layout.tsx
page.tsx
components
context
hooks
lib
node_modules
public
OUTLINE
TIMELINE
16 export default function RegisterPage() {
144 >
145 {showConfirmPassword ? (
146 <EyeOff className="h-4 w-4 text-gray-500" />
147) : (
148 <Eye className="h-4 w-4 text-gray-500" />
149)}
150 {showConfirmPassword ? "Hide password" : "Show password"}
151 </Button>
152
153
154
155 <Button type="submit" className="w-full" disabled={isLoading}>
156 {isLoading ? "Creating account..." : "Register"}
157 </Button>
158 </form>
159
160
161
162 Already have an account?{" "}
163 <Link href="/login" className="text-primary hover:underline">
164 Login
165 </Link>
166
167
168
169
170
171)
172 }
173
main* 0 0
Not Committed Yet Ln 57, Col 16 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Windsurf: {...} Prettier

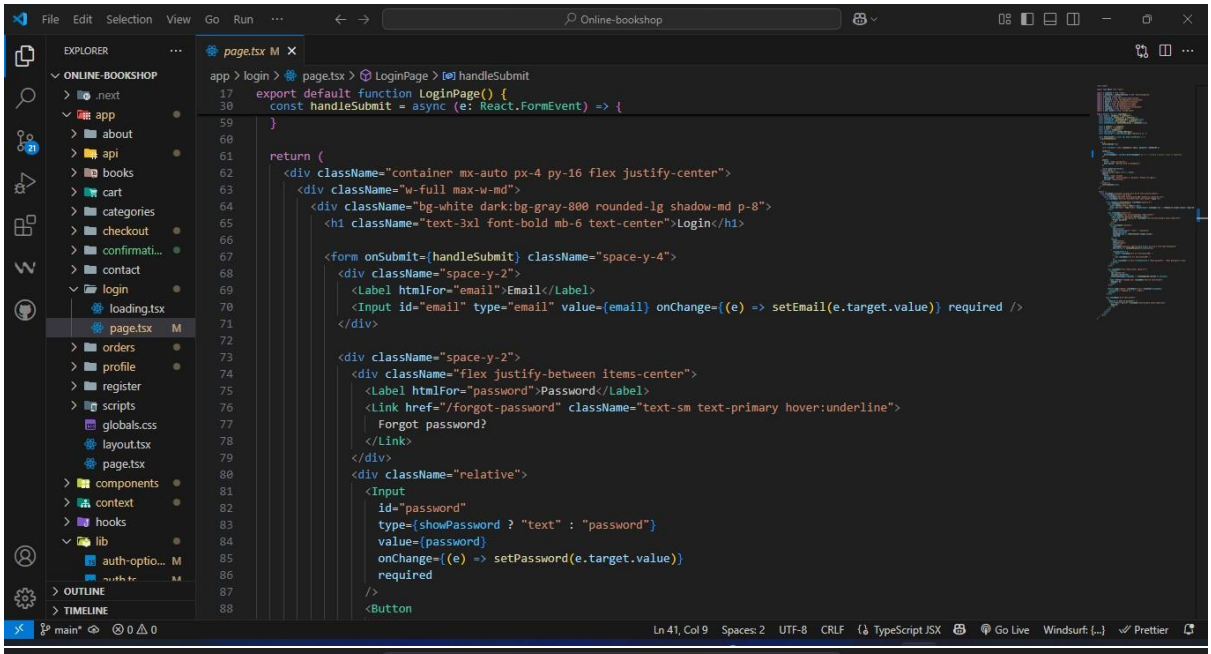
- Login page code screenshots



```
"use client"

import type React from "react"

import { useState } from "react"
import { useRouter, useSearchParams } from "next/navigation"
import Link from "next/link"
import { useAuth } from "@/context/auth-context"
import { useToast } from "@/components/ui/use-toast"
import { Button } from "@/components/ui/button"
import { Input } from "@/components/ui/input"
import { Label } from "@/components/ui/label"
import { Checkbox } from "@/components/ui/checkbox"
import { loginUser } from "@/lib/auth"
import { Eye, EyeOff } from "lucide-react"

Windsurf: Refactor | Explain | Generate JSDoc | X
export default function LoginPage() {
  const [email, setEmail] = useState("")
  const [password, setPassword] = useState("")
  const [rememberMe, setRememberMe] = useState(false)
  const [isLoading, setIsLoading] = useState(false)
  const [showPassword, setShowPassword] = useState(false)

  const { setUser } = useAuth()
  const { toast } = useToast()
  const router = useRouter()
  const searchParams = useSearchParams()
  const redirectUrl = searchParams.get("redirect") || "/"

  Windsurf: Refactor | Explain | Generate JSDoc | X
  const handleSubmit = async (e: React.FormEvent) => {
```



```
export default function LoginPage() {
Windsurf: Refactor | Explain | Generate JSDoc | X
  const handleSubmit = async (e: React.FormEvent) => {
    e.preventDefault()

    try {
      setIsLoading(true)

      const userData = await loginUser({ email, password, rememberMe })

      setUser({
        ...userData,
        profileImageUrl: userData.profileImageUrl || "", // Provide a default value if undefined
      })

      toast({
        title: "Login successful",
        description: "Welcome back to BookHaven!",
      })

      router.push(redirectUrl)
    } catch (error) {
      console.error("Login error:", error)
      toast({
        title: "Login failed",
        description: "Invalid email or password. Please try again.",
        variant: "destructive",
      })
    } finally {
      setIsLoading(false)
    }
  }
```

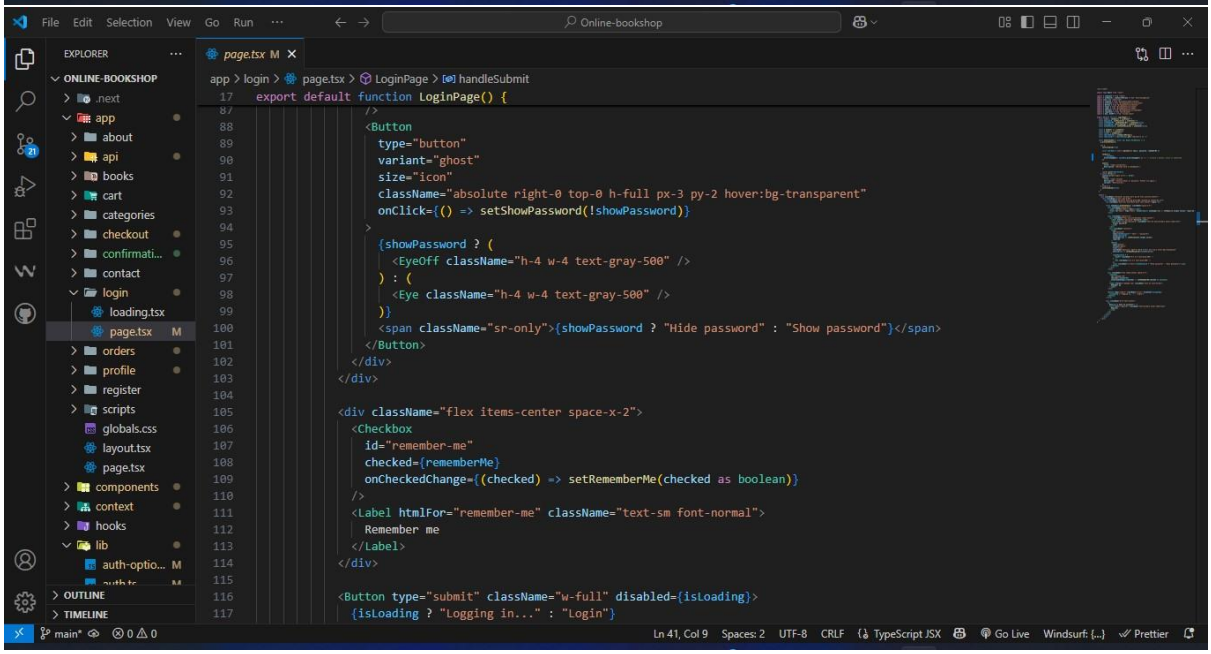

File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > login > page.tsx > LoginPage > handleSubmit
ONLINE-BOOKSHOP
export default function LoginPage() {
const handleSubmit = async (e: React.FormEvent) => {
}
return (

<h1 className="text-3xl font-bold mb-6 text-center">Login</h1>
<form onSubmit={handleSubmit} className="space-y-4">

<Label htmlFor="email">Email</Label>
<Input id="email" type="email" value={email} onChange={(e) => setEmail(e.target.value)} required />

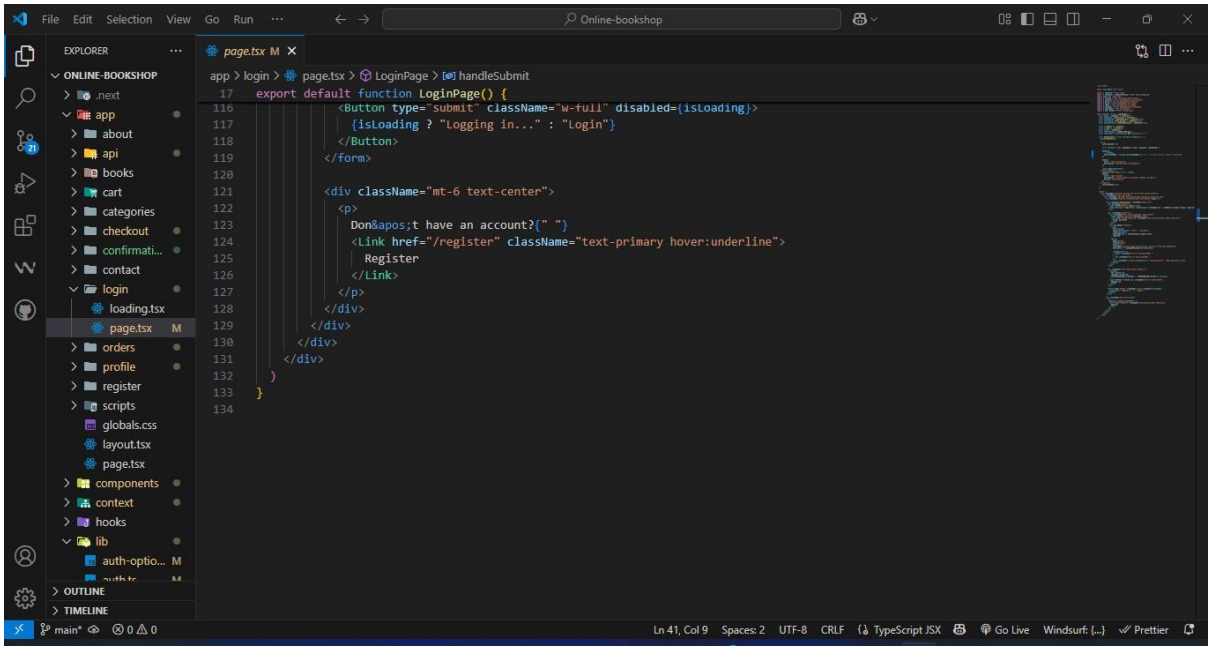
<Label htmlFor="password">Password</Label>
<Link href="/forgot-password" className="text-sm text-primary hover:underline">
Forgot password?
</Link>

<Input
id="password"
type={showPassword ? "text" : "password"}
value={password}
onChange={(e) => setPassword(e.target.value)}
required
/>
<Button
Ln 41, Col 9 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > login > page.tsx > LoginPage > handleSubmit
ONLINE-BOOKSHOP
export default function LoginPage() {
/>
<Button
type="button"
variant="ghost"
size="icon"
className="absolute right-0 top-0 h-full px-3 py-2 hover:bg-transparent"
onClick={() => setShowPassword(!showPassword)}
>
{showPassword ? (
<EyeOff className="h-4 w-4 text-gray-500" />
) : (
<Eye className="h-4 w-4 text-gray-500" />
)}
{showPassword ? "Hide password" : "Show password"}
</Button>

<Checkbox
id="remember-me"
checked={rememberMe}
onCheckedChange={(checked) => setRememberMe(checked as boolean)}
/>
<Label htmlFor="remember-me" className="text-sm font-normal">
Remember me
</Label>

<Button type="submit" className="w-full" disabled={isLoading}>
{isLoading ? "Logging in..." : "Login"}
Ln 41, Col 9 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx M
app > login > page.tsx > LoginPage > handleSubmit
ONLINE-BOOKSHOP
.next
app
about
api
books
cart
categories
checkout
confirmati...
contact
login
loading.tsx
page.tsx M
orders
profile
register
scripts
globals.css
layout.tsx
page.tsx
components
context
hooks
lib
auth-optio... M
OUTLINE
TIMELINE
export default function LoginPage() {
<Button type="submit" className="w-full" disabled={isLoading}>
{isLoading ? "Logging in..." : "Login"}
</Button>
</form>

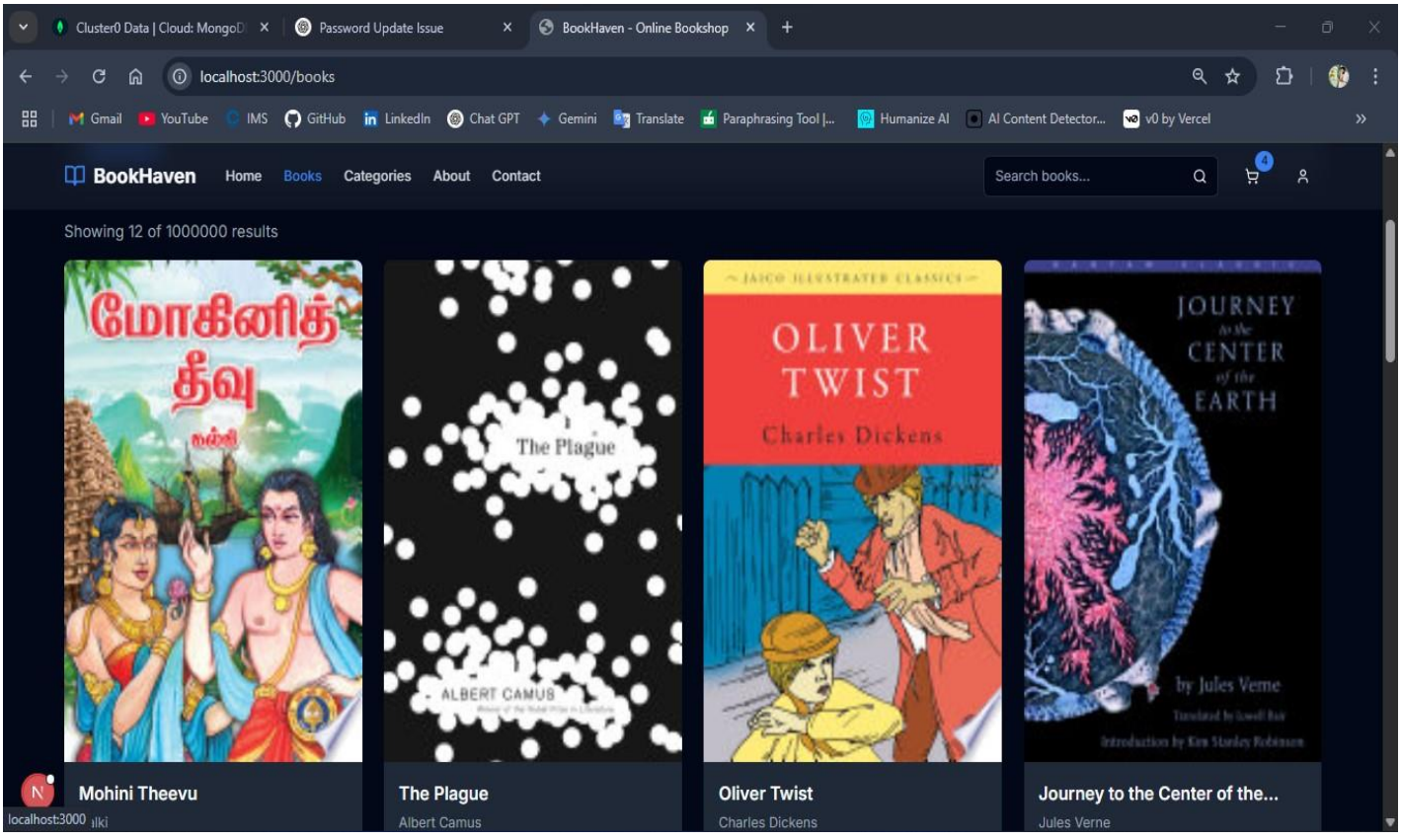
Don't have an account?{" "}
<Link href="/register" className="text-primary hover:underline">
Register
</Link>

)
}
main*
Ln 41, Col 9
Spaces: 2
UTF-8
CRLF
TypeScript JSX
Go Live
Windsurf: {...}
Prettier

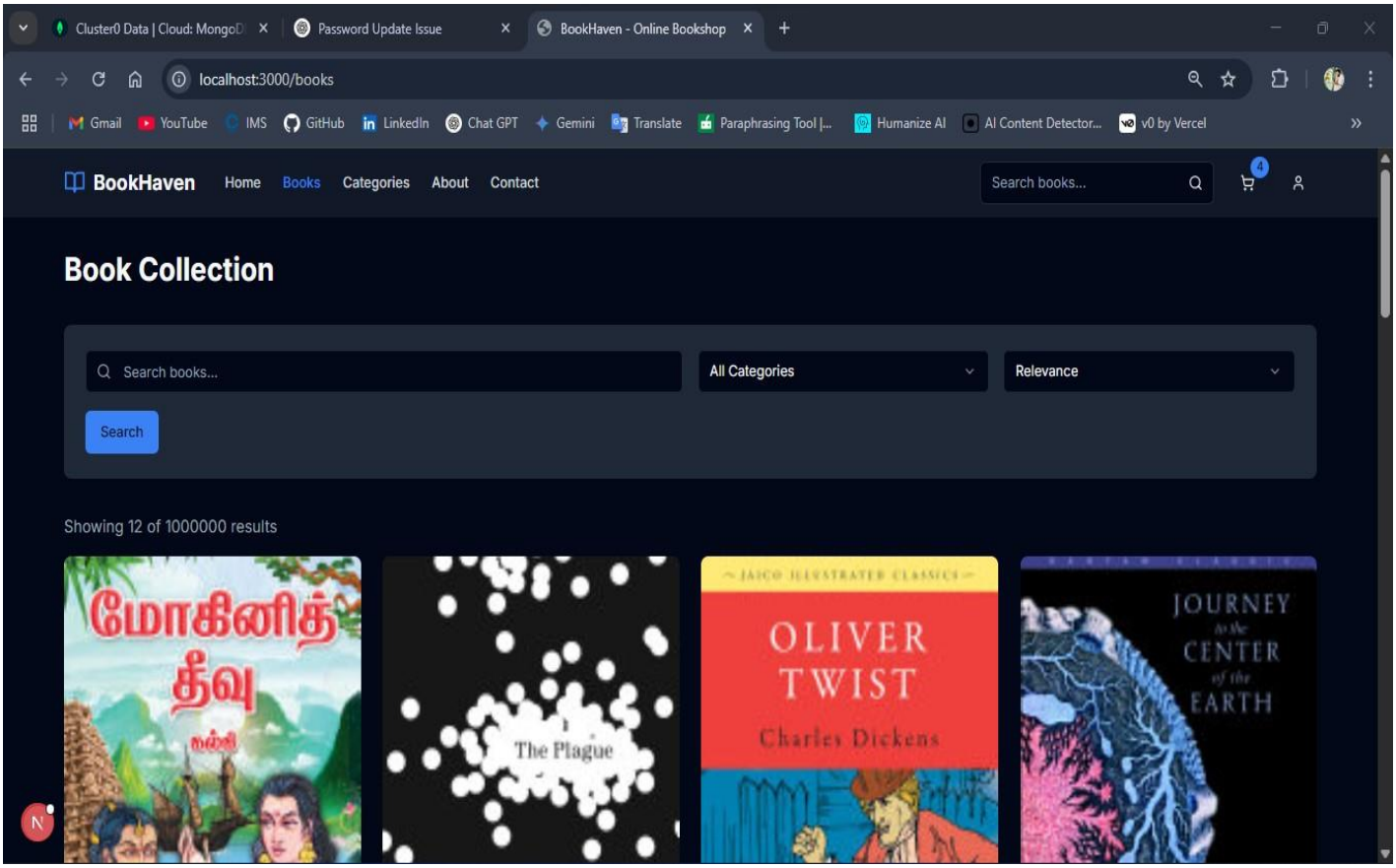
- Authentication success/error messages screenshots

1.2. Book Listing and Search

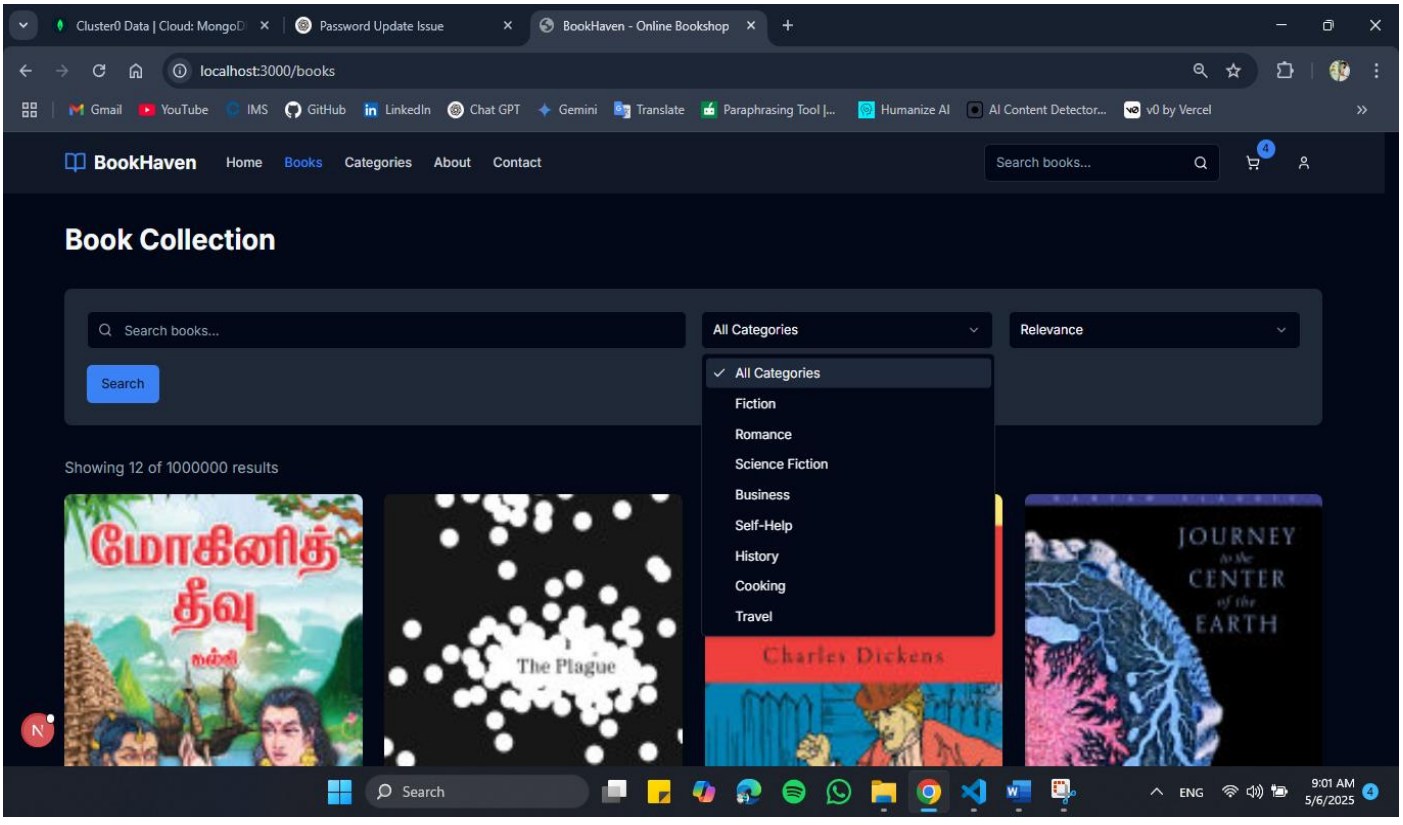
- Main book listing page screenshot

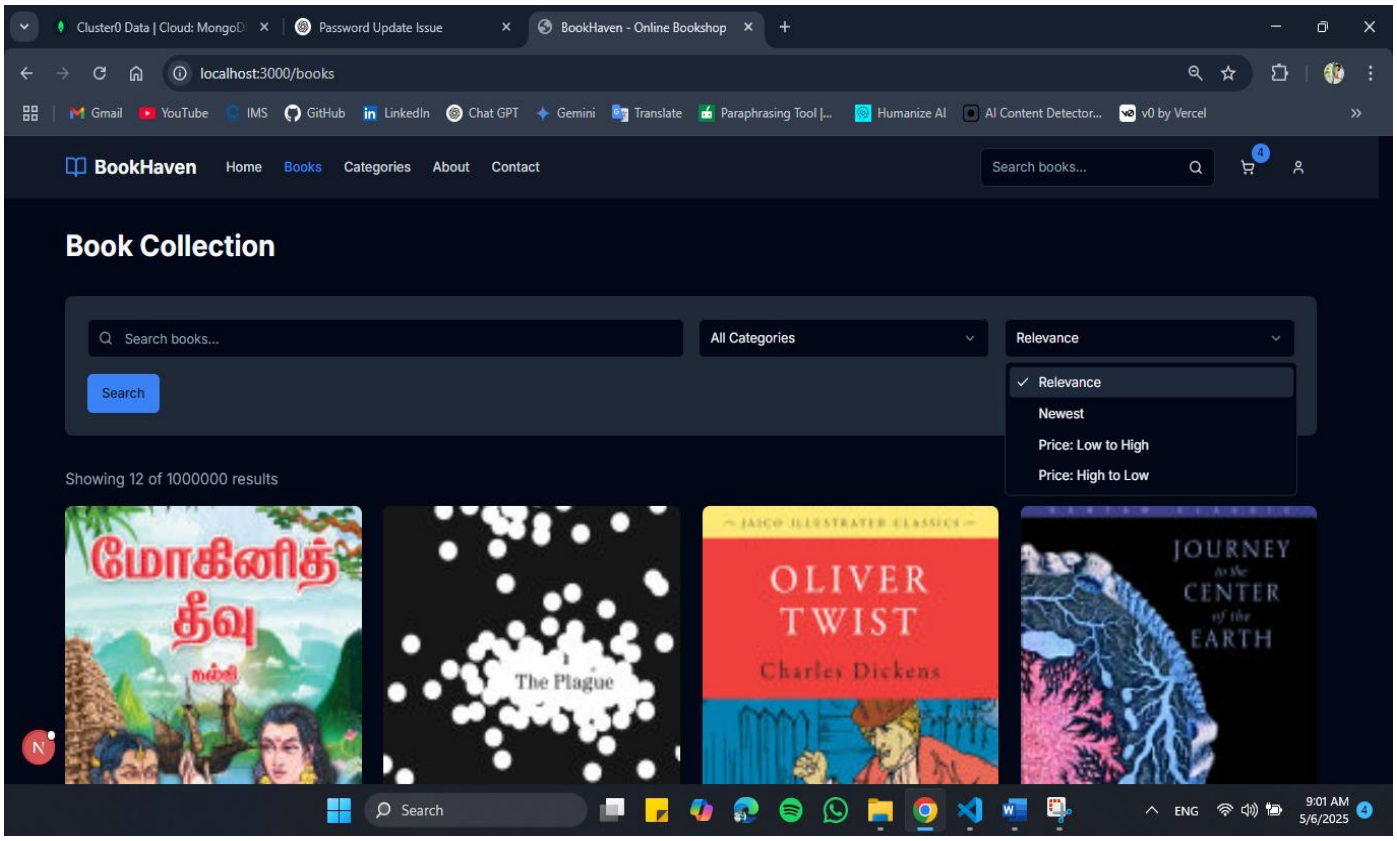


- Search functionality interface screenshot

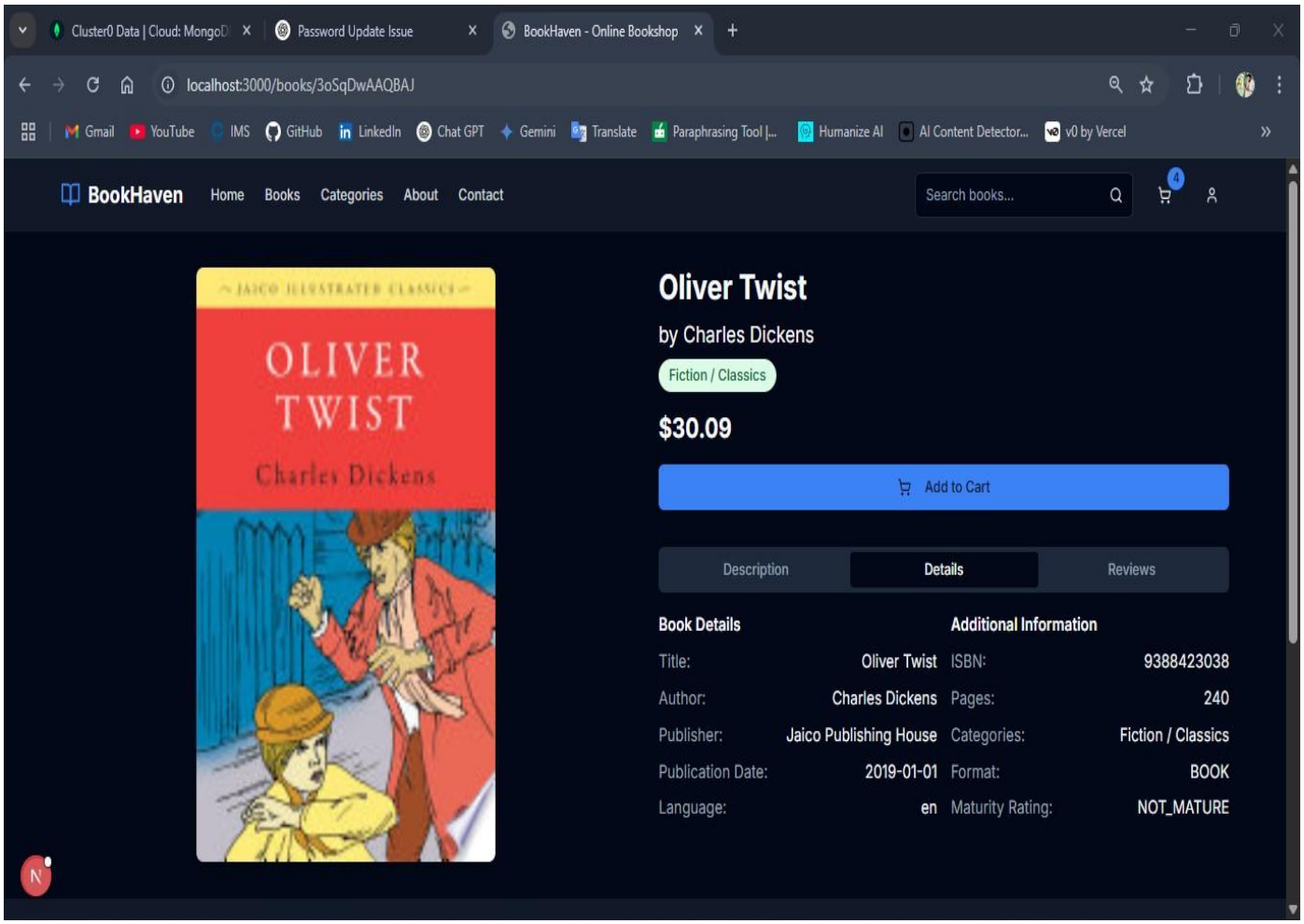


- Filter/Sort options screenshot

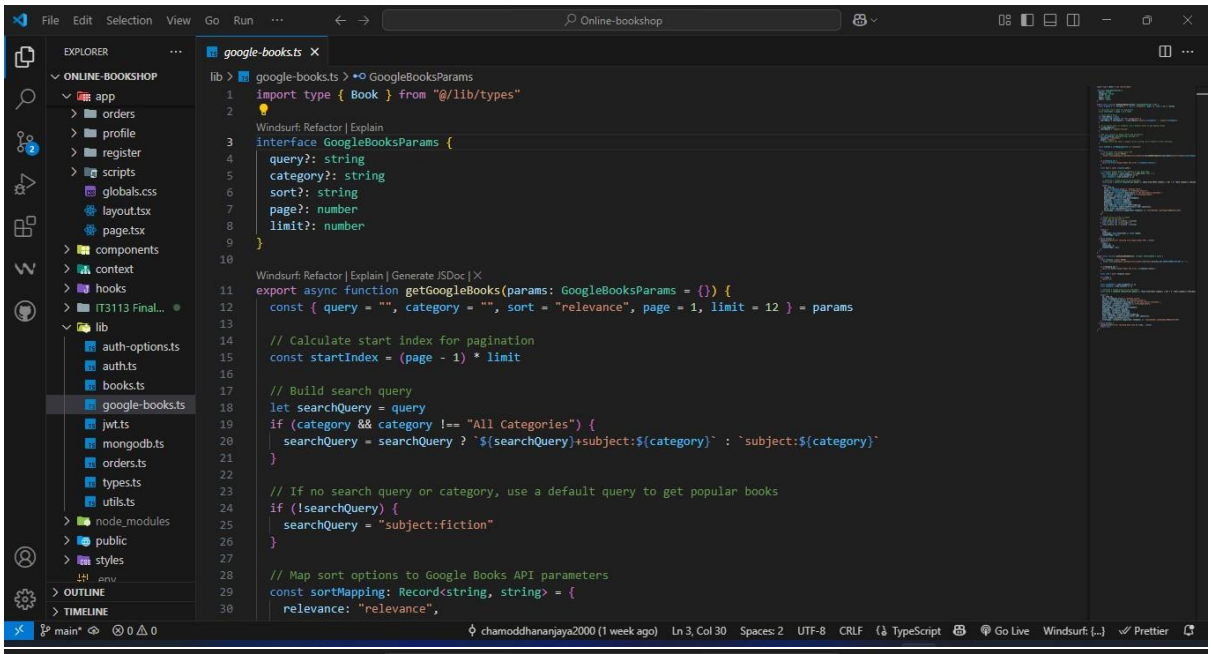




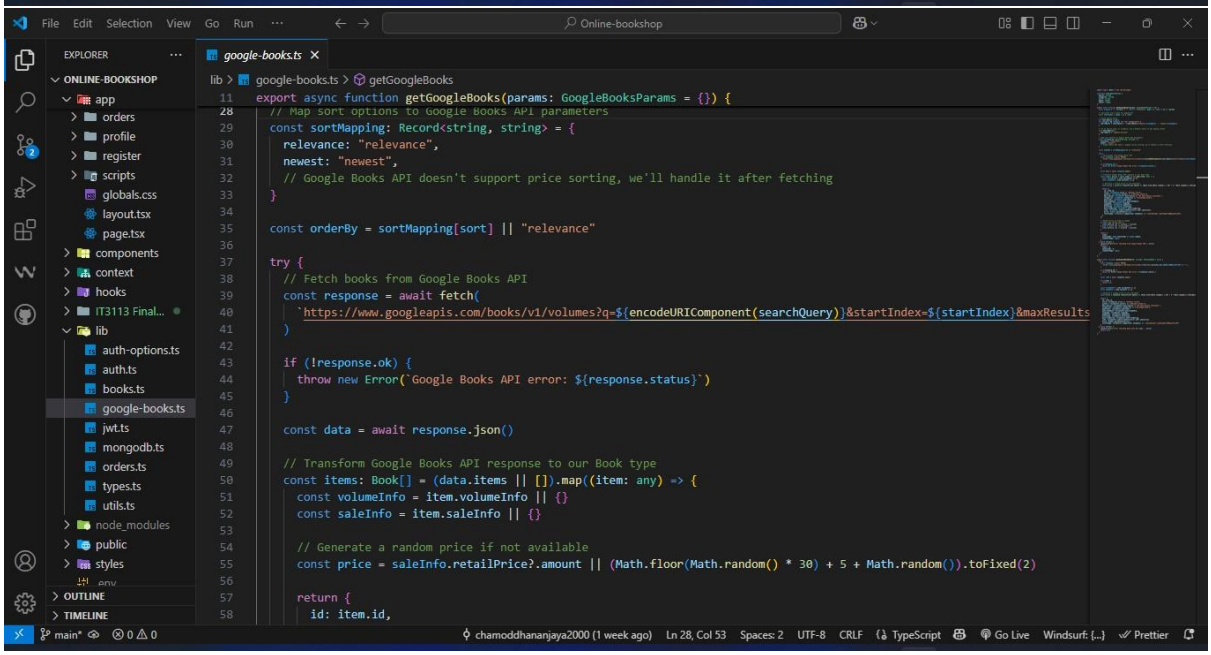
➢ Book detail's view screenshot



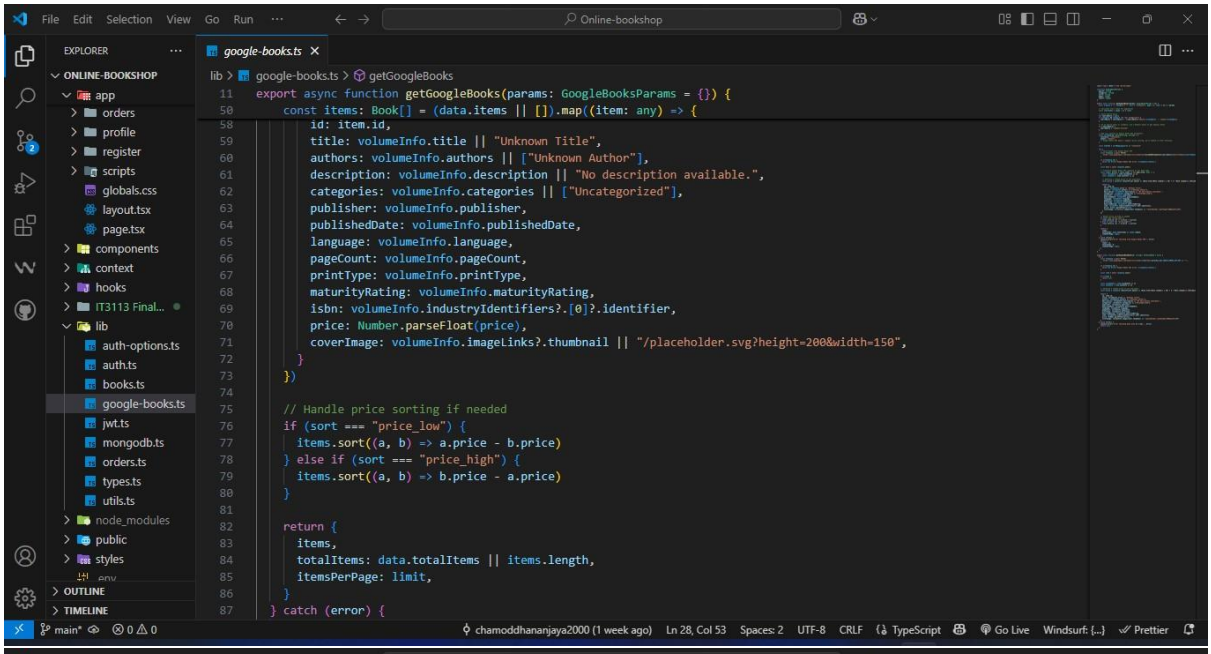
- Code screenshots for API integration



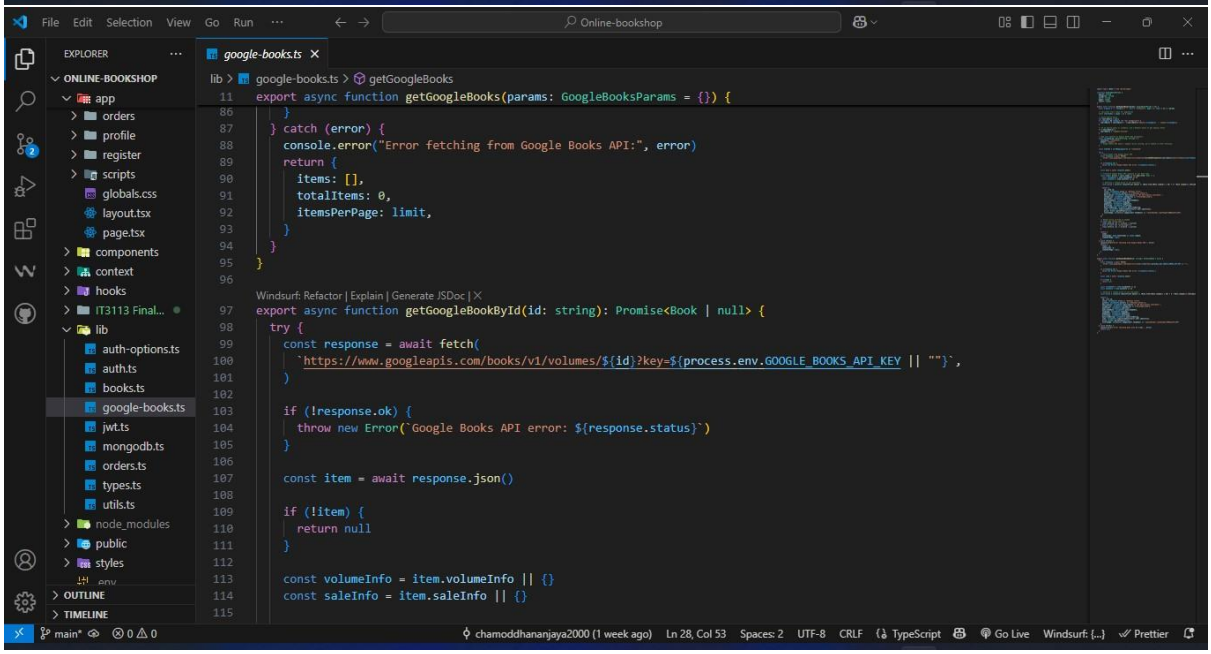
```
import type { Book } from "@/lib/types"

Windsurf: Refactor | Explain
interface GoogleBooksParams {
  query?: string
  category?: string
  sort?: string
  page?: number
  limit?: number
}

Windsurf: Refactor | Explain | Generate JSDoc |
export async function getGoogleBooks(params: GoogleBooksParams = {}) {
  const { query = "", category = "", sort = "relevance", page = 1, limit = 12 } = params

  // Calculate start index for pagination
  const startIndex = (page - 1) * limit

  // Build search query
  let searchQuery = query
  if (category && category !== "All Categories") {
    searchQuery = searchQuery ? `${searchQuery}+subject:${category}` : `subject:${category}`
  }

  // If no search query or category, use a default query to get popular books
  if (!searchQuery) {
    searchQuery = "subject:fiction"
  }

  // Map sort options to Google Books API parameters
  const sortMapping: Record<string, string> = {
    relevance: "relevance",
```



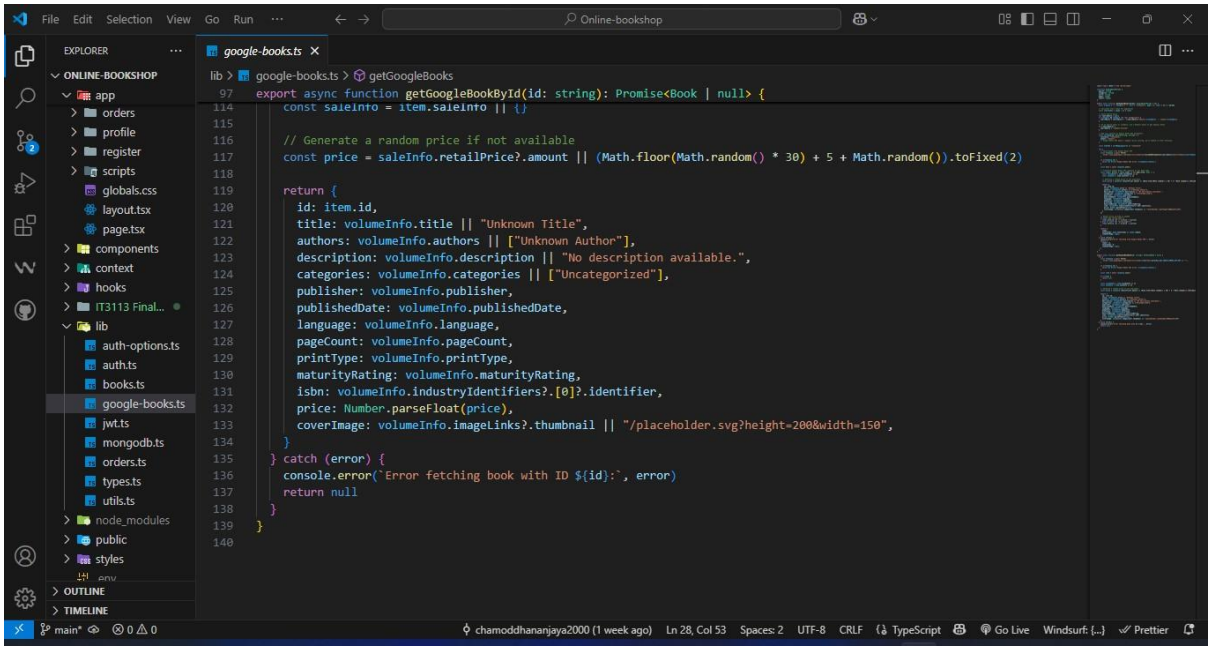
```
export async function getGoogleBooks(params: GoogleBooksParams = {}) {
  // Map sort options to Google Books API parameters
  const sortMapping: Record<string, string> = {
    relevance: "relevance",
    newest: "newest",
    // Google Books API doesn't support price sorting, we'll handle it after fetching
  }

  const orderBy = sortMapping[sort] || "relevance"

  try {
    // Fetch books from Google Books API
    const response = await fetch(
      `https://www.googleapis.com/books/v1/volumes?q=${encodeURIComponent(searchQuery)}&startIndex=${startIndex}&maxResults
    )

    if (!response.ok) {
      throw new Error(`Google Books API error: ${response.status}`)
    }

    const data = await response.json()

    // Transform Google Books API response to our Book type
    const items: Book[] = (data.items || []).map((item: any) => {
      const volumeInfo = item.volumeInfo || {}
      const saleInfo = item.saleInfo || {}

      // Generate a random price if not available
      const price = saleInfo.retailPrice?.amount || (Math.floor(Math.random() * 30) + 5 + Math.random()).toFixed(2)

      return {
        id: item.id,
```



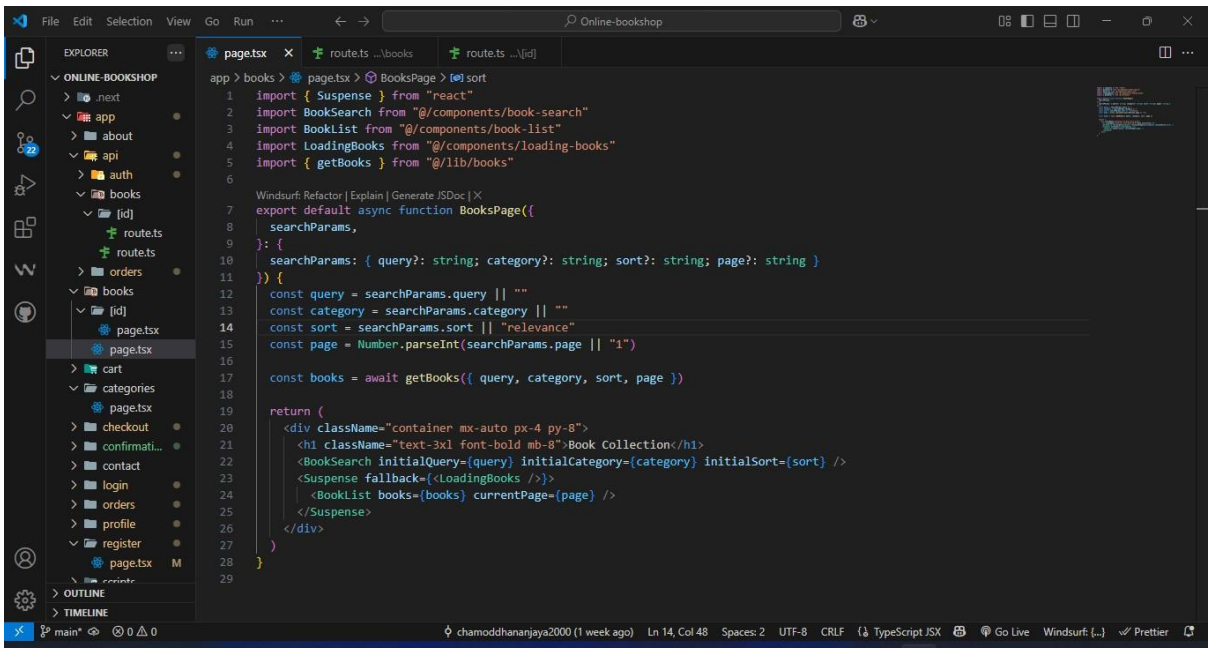
```
export async function getGoogleBooks(params: GoogleBooksParams = {}) {
    const items: Book[] = (data.items || []).map((item: any) => {
        id: item.id,
        title: volumeInfo.title || "Unknown Title",
        authors: volumeInfo.authors || ["Unknown Author"],
        description: volumeInfo.description || "No description available.",
        categories: volumeInfo.categories || ["Uncategorized"],
        publisher: volumeInfo.publisher,
        publishedDate: volumeInfo.publishedDate,
        language: volumeInfo.language,
        pageCount: volumeInfo.pageCount,
        printType: volumeInfo.printType,
        maturityRating: volumeInfo.maturityRating,
        isbn: volumeInfo.industryIdentifiers?.[0]?.identifier,
        price: Number.parseFloat(price),
        coverImage: volumeInfo.imageLinks?.thumbnail || "/placeholder.svg?height=200&width=150",
      }
    })

    // Handle price sorting if needed
    if (sort === "price_low") {
      items.sort((a, b) => a.price - b.price)
    } else if (sort === "price_high") {
      items.sort((a, b) => b.price - a.price)
    }

    return {
      items,
      totalItems: data.totalItems || items.length,
      itemsPerPage: limit,
    }
  } catch (error) {
```

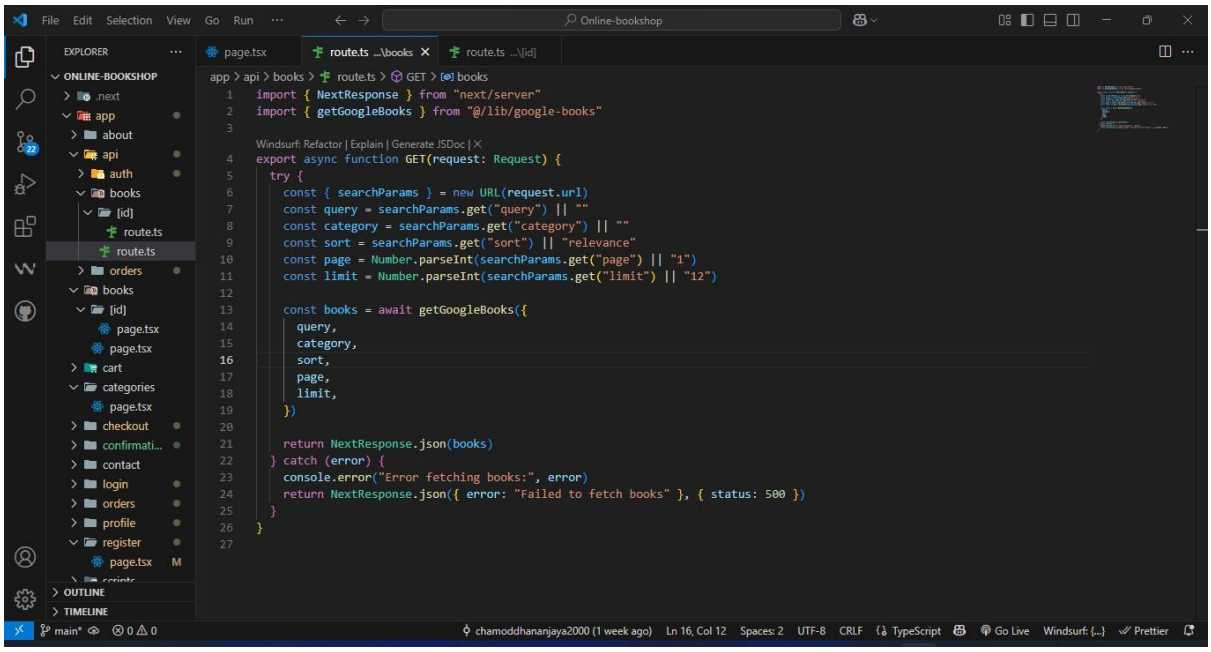


```
export async function getGoogleBooks(params: GoogleBooksParams = {}) {
    }
  } catch (error) {
    console.error("Error fetching from Google Books API:", error)
    return {
      items: [],
      totalItems: 0,
      itemsPerPage: limit,
    }
  }
}

Windsurf: Refactor | Explain | Generate JSDoc | ×
export async function getGoogleBookById(id: string): Promise<Book | null> {
  try {
    const response = await fetch(
      `https://www.googleapis.com/books/v1/volumes/${id}?key=${process.env.GOOGLE_BOOKS_API_KEY || ""}`,
    )

    if (!response.ok) {
      throw new Error(`Google Books API error: ${response.status}`)
    }

    const item = await response.json()

    if (!item) {
      return null
    }

    const volumeInfo = item.volumeInfo || {}
    const saleInfo = item.saleInfo || {}
```



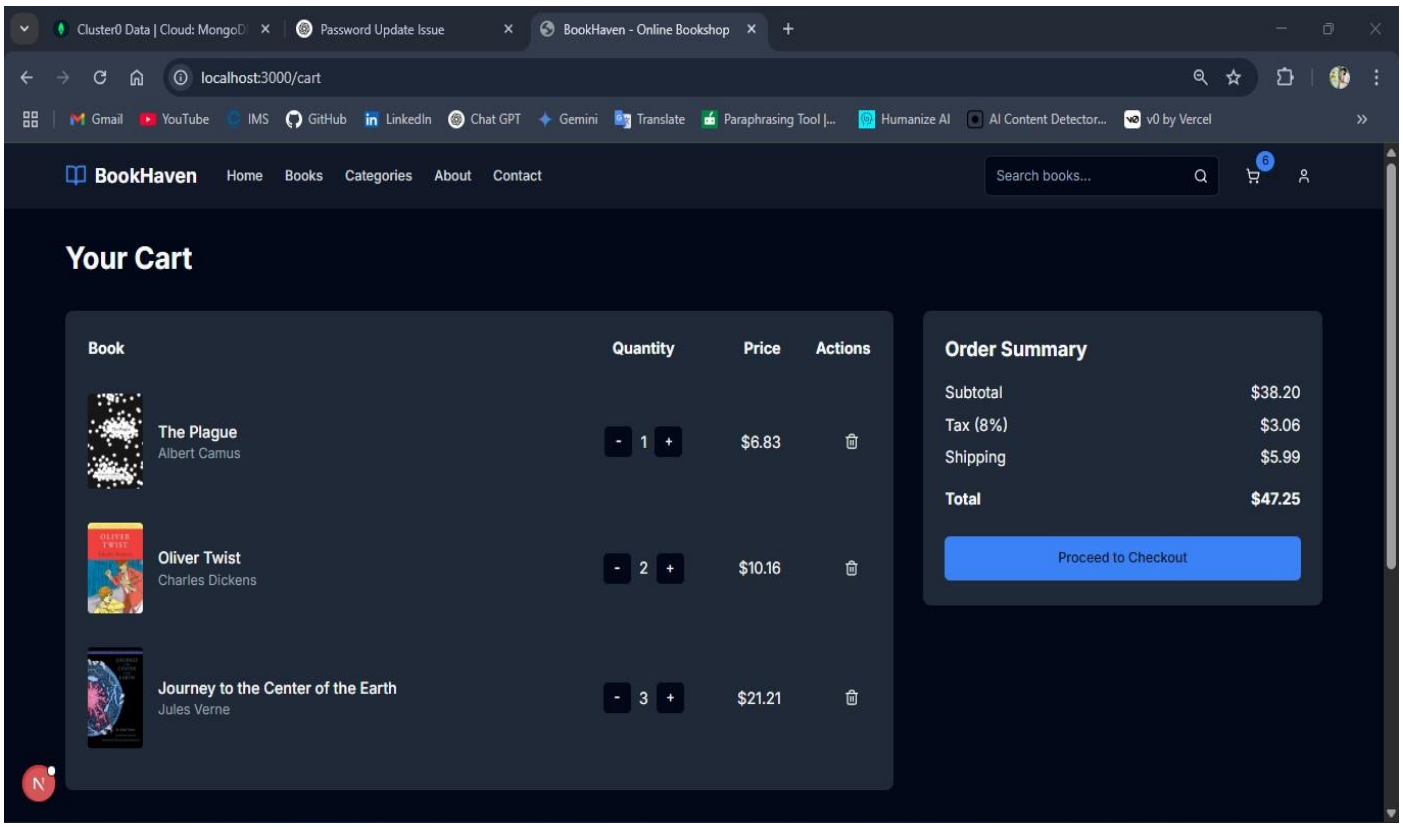
➢ Search implementation code screenshots



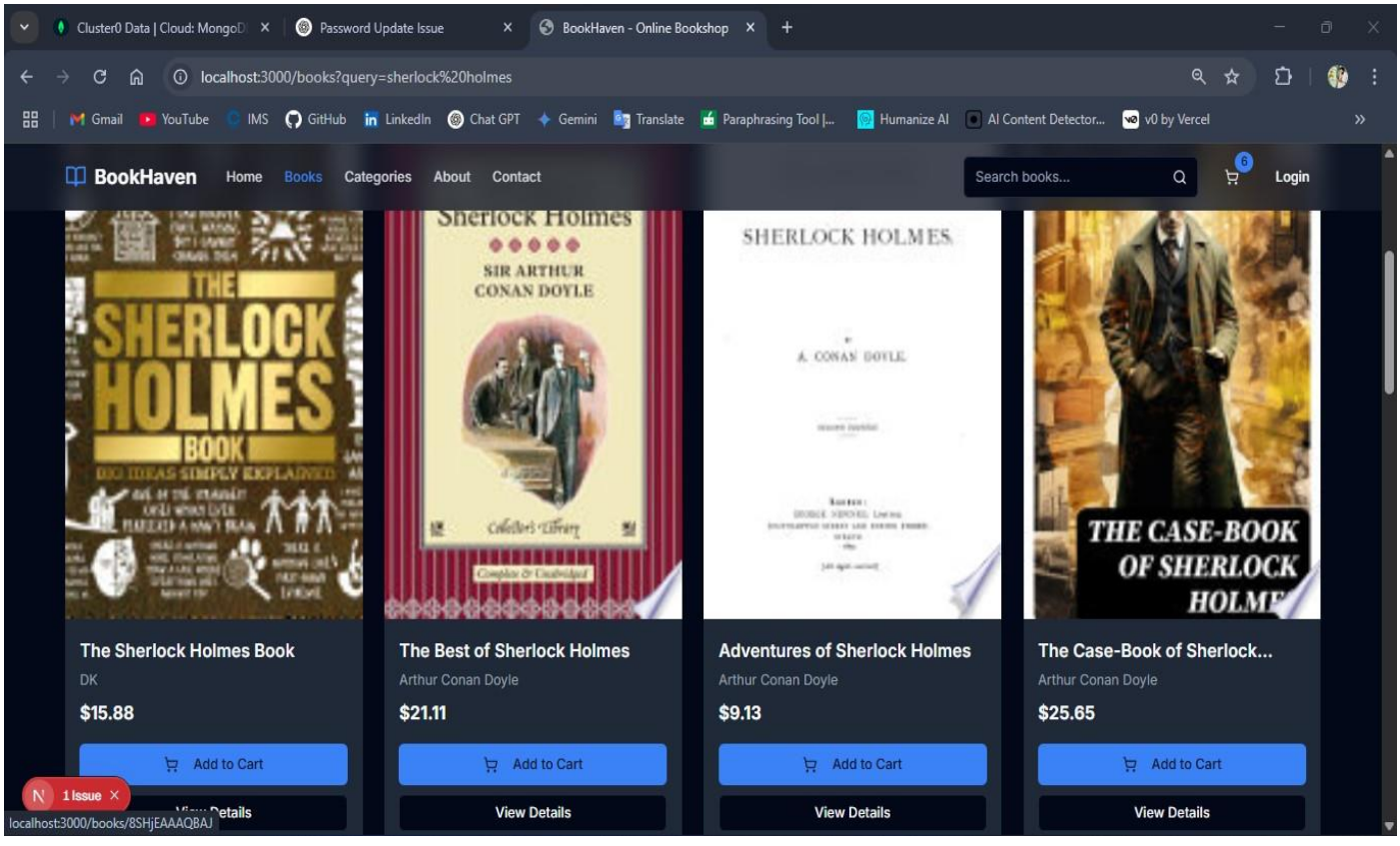

File Edit Selection View Go Run
Online-bookshop
EXPLORER
ONLINE-BOOKSHOP
page.tsx
route.ts ...\books
route.ts ...\[id]
app > api > books > route.ts > GET > books
Windsurf: Refactor | Explain | Generate JSDoc |
import { NextResponse } from "next/server"
import { getGoogleBooks } from "@/lib/google-books"
export async function GET(request: Request) {
try {
const { searchParams } = new URL(request.url)
const query = searchParams.get("query") || ""
const category = searchParams.get("category") || ""
const sort = searchParams.get("sort") || "relevance"
const page = Number.parseInt(searchParams.get("page") || "1")
const limit = Number.parseInt(searchParams.get("limit") || "12")
const books = await getGoogleBooks({
query,
category,
sort,
page,
limit,
})
return NextResponse.json(books)
} catch (error) {
console.error("Error fetching books:", error)
return NextResponse.json({ error: "Failed to fetch books" }, { status: 500 })
}
}
OUTLINE
TIMELINE
main*
chamoddhananjaya2000 (1 week ago)
Ln 16, Col 12
Spaces: 2
UTF-8
CRLF
TypeScript
Go Live
Windsurf: {...}
Prettier

1.3. Shopping Cart

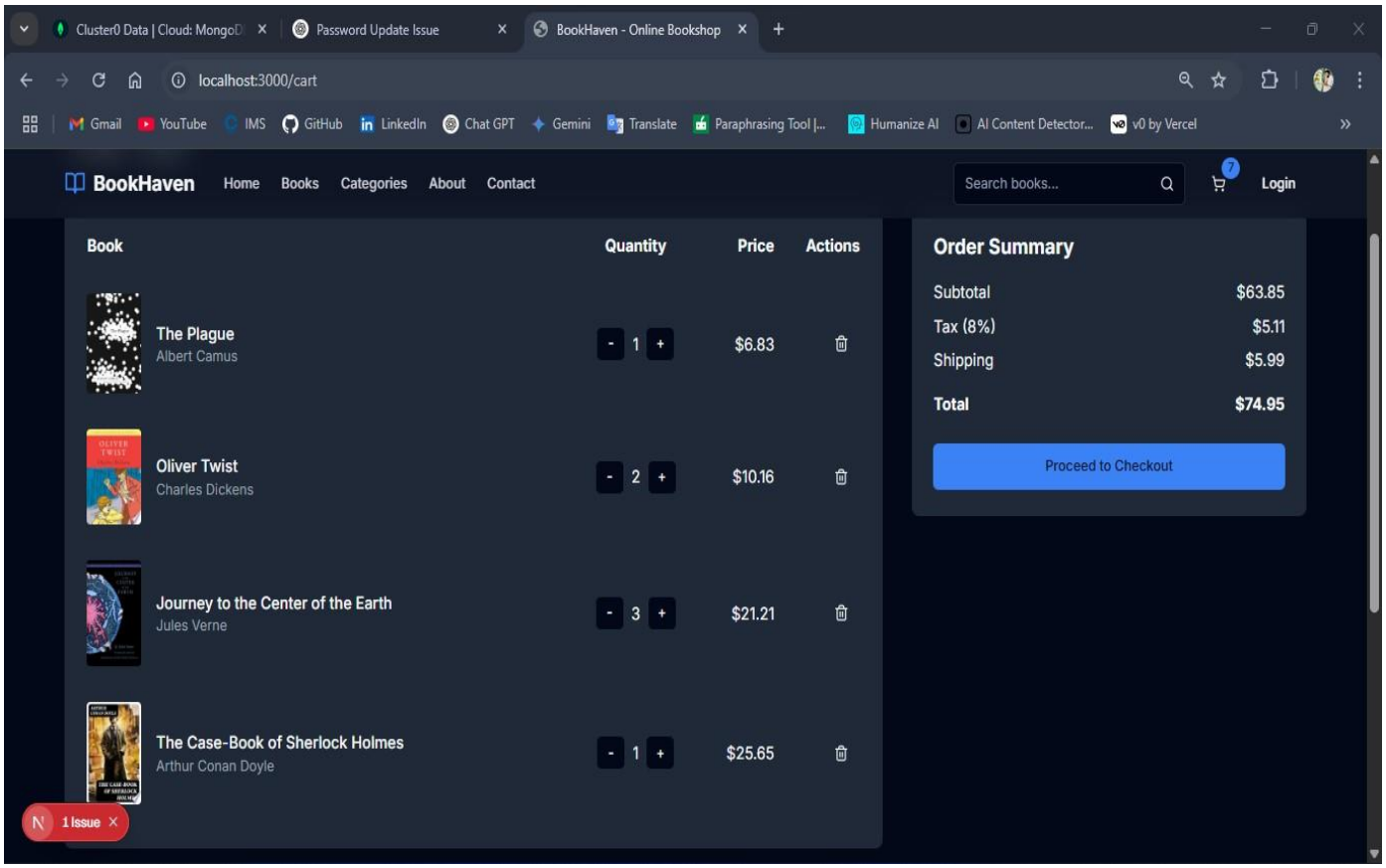
- Cart view screenshot



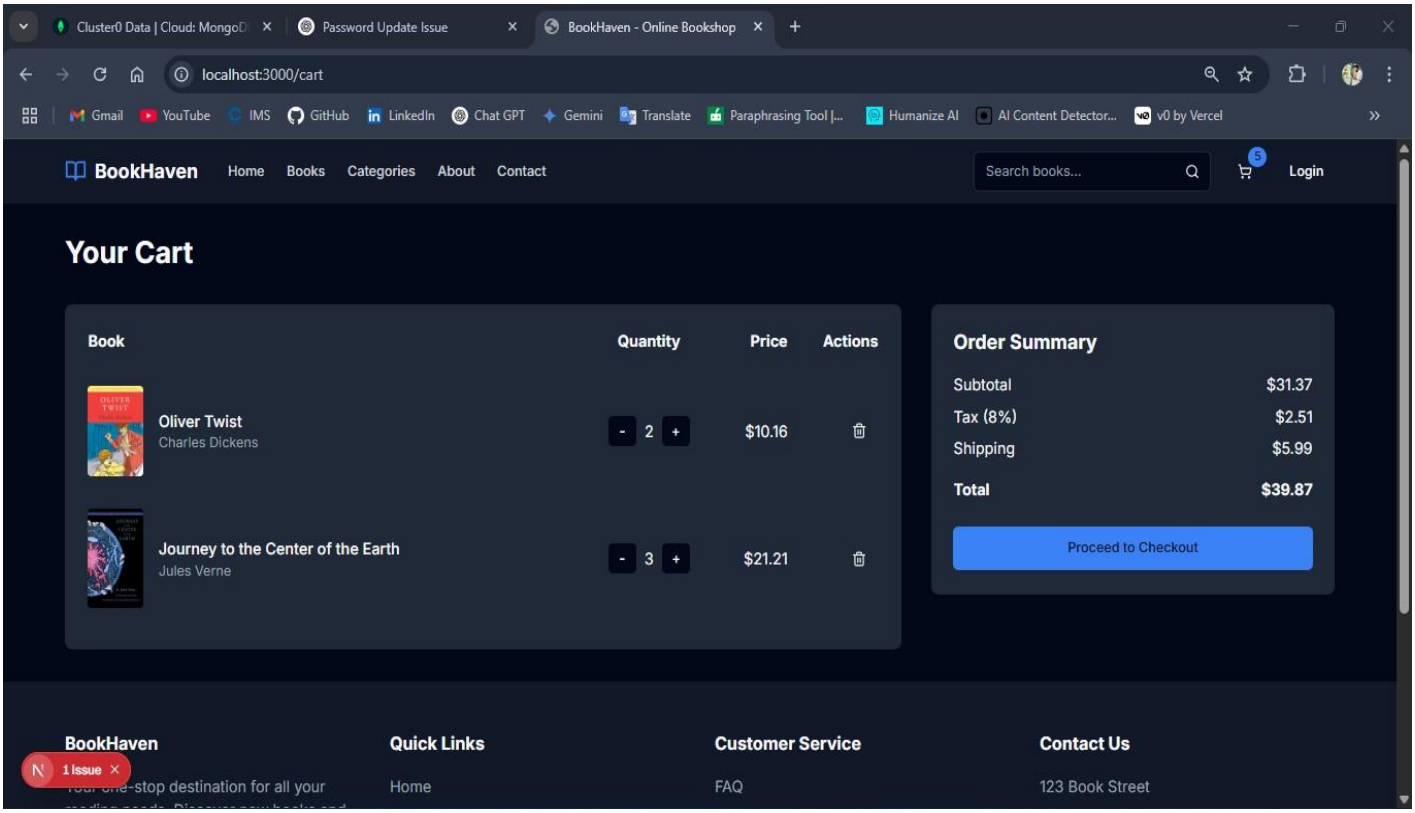
- Add to cart functionality screenshot



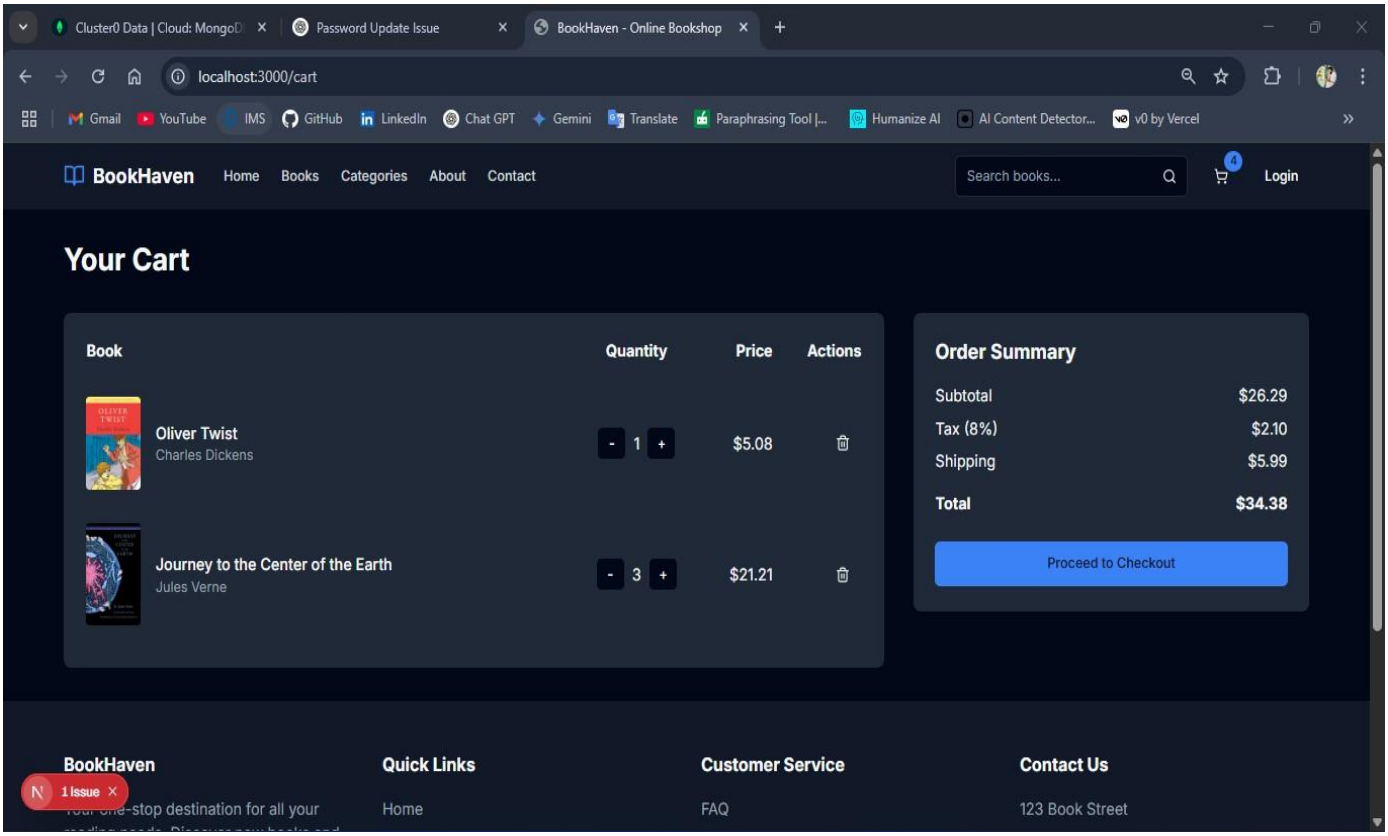
- Update quantity interface screenshot



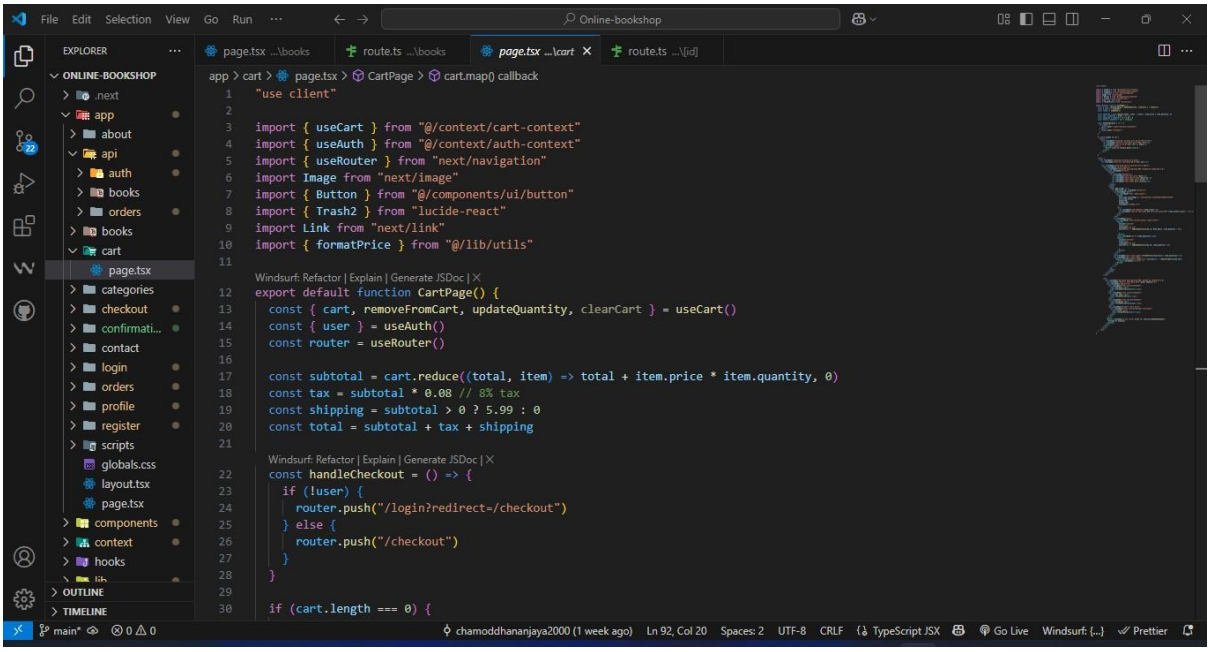
- Remove items interface screenshot

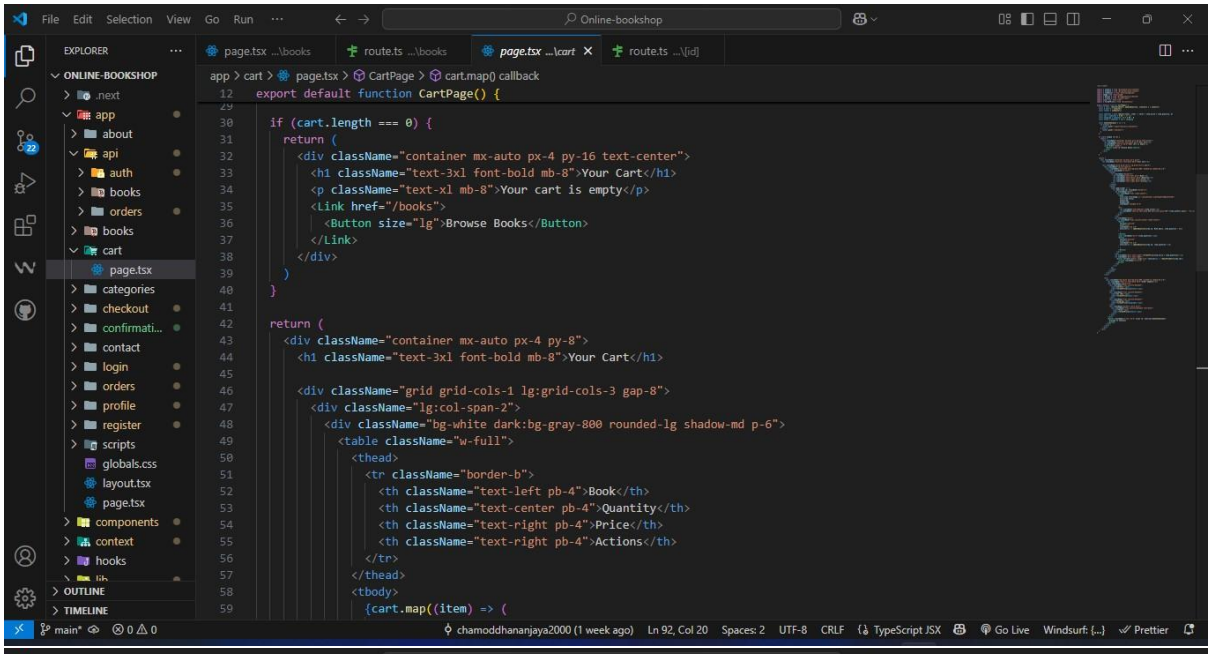


- Cart total calculation screenshot

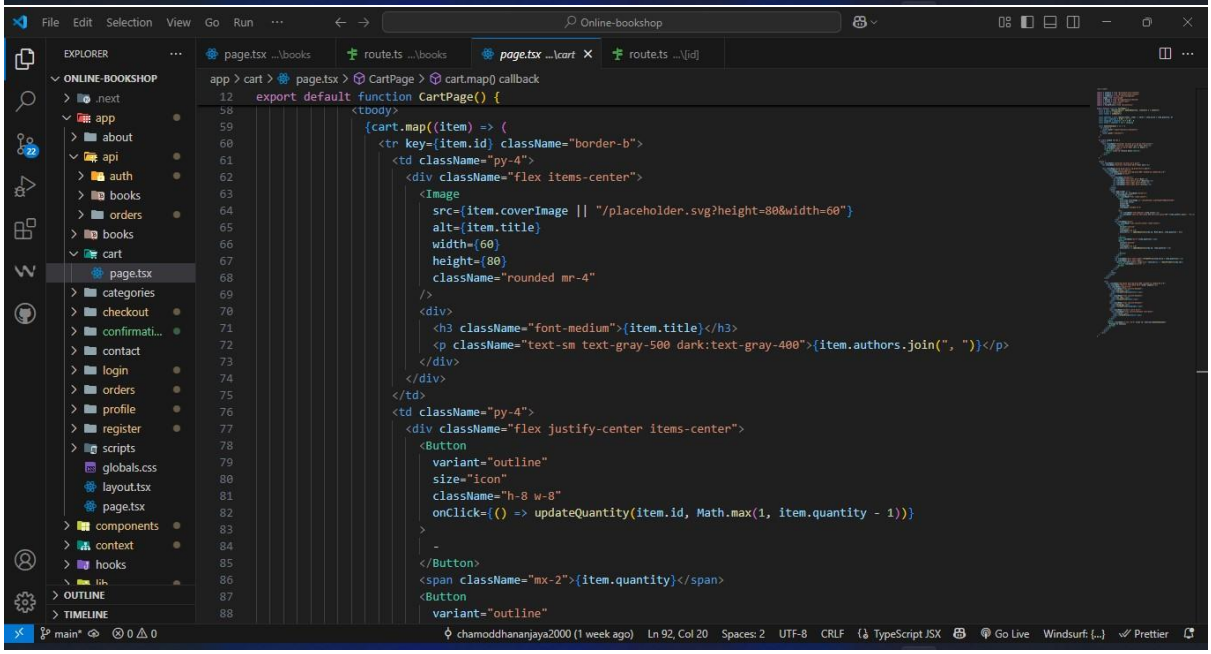


- Related code screenshots for cart management

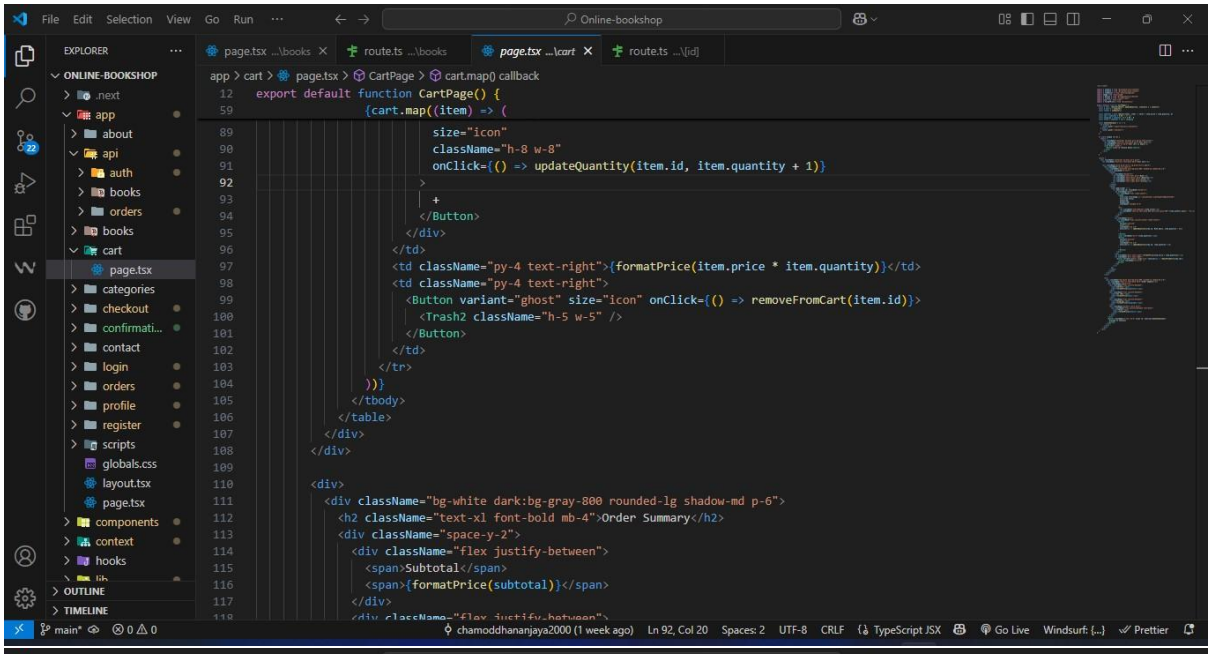




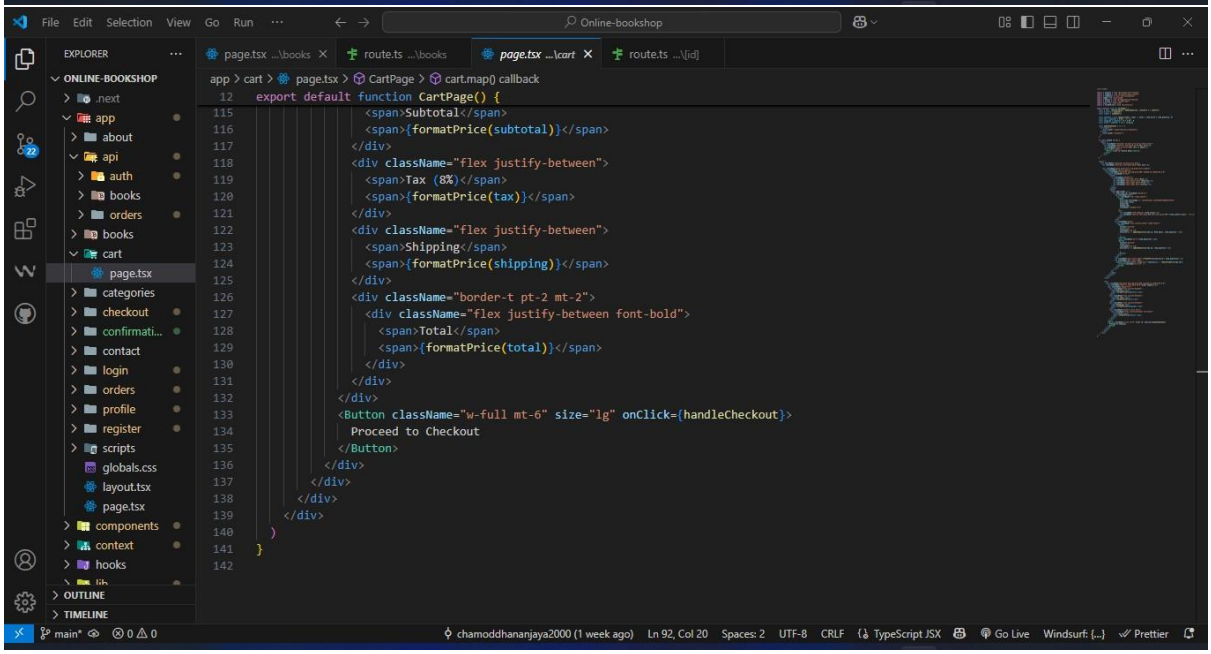
```
export default function CartPage() {

  if (cart.length === 0) {
    return (
      <div className="container mx-auto px-4 py-16 text-center">
        <h1 className="text-3xl font-bold mb-8">Your Cart</h1>
        <p className="text-xl mb-8">Your cart is empty</p>
        <Link href="/books">
          <Button size="lg">Browse Books</Button>
        </Link>
      </div>
    )
  }

  return (
    <div className="container mx-auto px-4 py-8">
      <h1 className="text-3xl font-bold mb-8">Your Cart</h1>

      <div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
        <div className="lg:col-span-2">
          <div className="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6">
            <table className="w-full">
              <thead>
                <tr className="border-b">
                  <th className="text-left pb-4">Book</th>
                  <th className="text-center pb-4">Quantity</th>
                  <th className="text-right pb-4">Price</th>
                  <th className="text-right pb-4">Actions</th>
                </tr>
              </thead>
              <tbody>
                {cart.map((item) => (
```



```
export default function CartPage() {
              <tbody>
                {cart.map((item) => (
                  <tr key={item.id} className="border-b">
                    <td className="py-4">
                      <div className="flex items-center">
                        <Image
                          src={item.coverImage || "/placeholder.svg?height=80&width=60"}
                          alt={item.title}
                          width={60}
                          height={80}
                          className="rounded mr-4"
                        />
                        <div>
                          <h3 className="font-medium">{item.title}</h3>
                          <p className="text-sm text-gray-500 dark:text-gray-400">{item.authors.join(", ")}</p>
                        </div>
                      </div>
                    </td>
                    <td className="py-4">
                      <div className="flex justify-center items-center">
                        <Button
                          variant="outline"
                          size="icon"
                          className="h-8 w-8"
                          onClick={() => updateQuantity(item.id, Math.max(1, item.quantity - 1))}
                        >
                          -
                        </Button>
                        <span className="mx-2">{item.quantity}</span>
                        <Button
                          variant="outline"
```



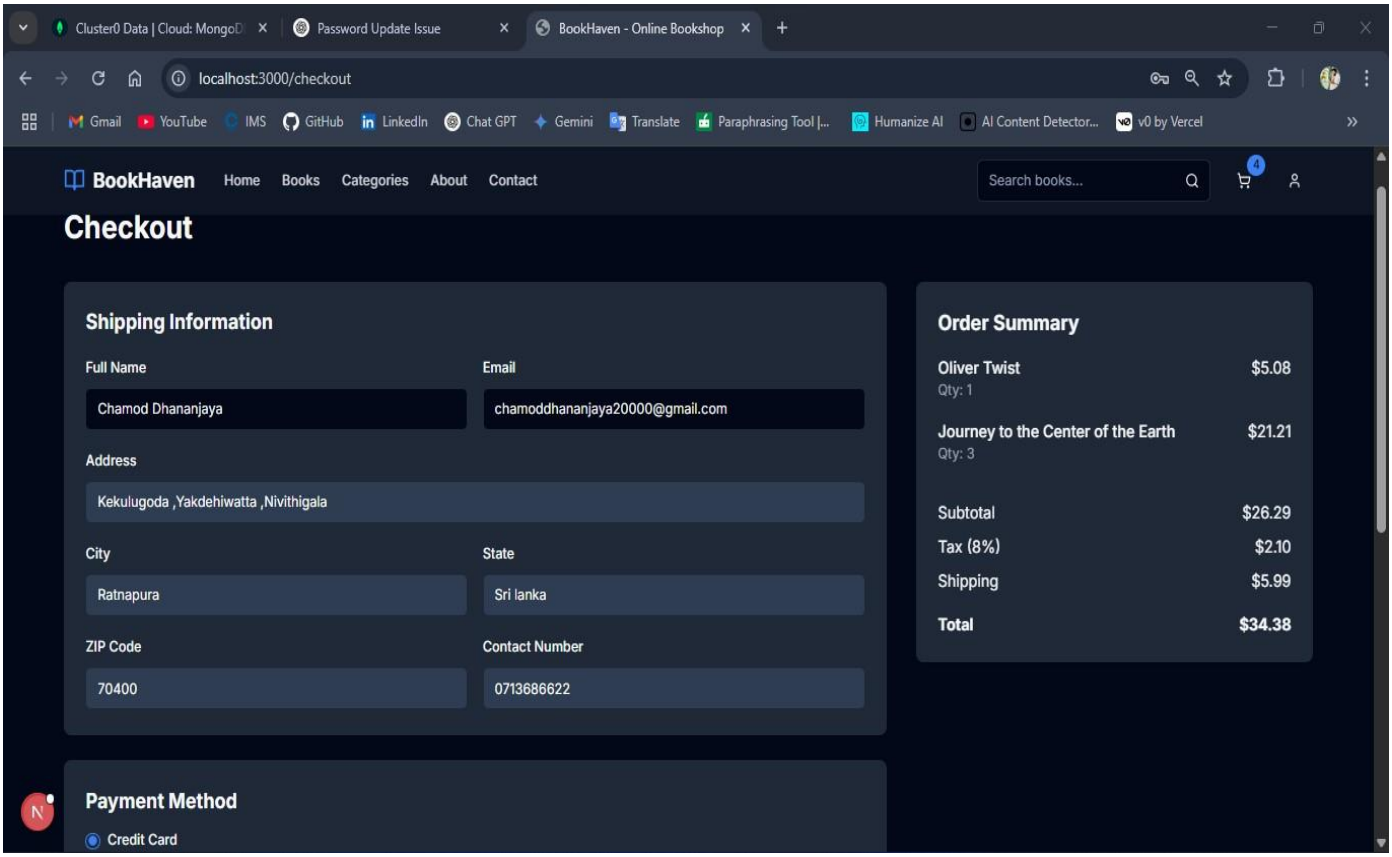
```
export default function CartPage() {
                {cart.map((item) => (
                                size="icon"
                                className="h-8 w-8"
                                onClick={() => updateQuantity(item.id, item.quantity + 1)}
                              >
                                +
                              </Button>
                            </div>
                          </td>
                          <td className="py-4 text-right">{formatPrice(item.price * item.quantity)}</td>
                          <td className="py-4 text-right">
                            <Button variant="ghost" size="icon" onClick={() => removeFromCart(item.id)}>
                              <Trash2 className="h-5 w-5" />
                            </Button>
                          </td>
                        </tr>
                      ))}
                    </tbody>
                  </table>
                </div>
              </div>

              <div>
                <div className="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6">
                  <h2 className="text-xl font-bold mb-4">Order Summary</h2>
                  <div className="space-y-2">
                    <div className="flex justify-between">
                      <span>Subtotal</span>
                      <span>{formatPrice(subtotal)}</span>
                    </div>
```



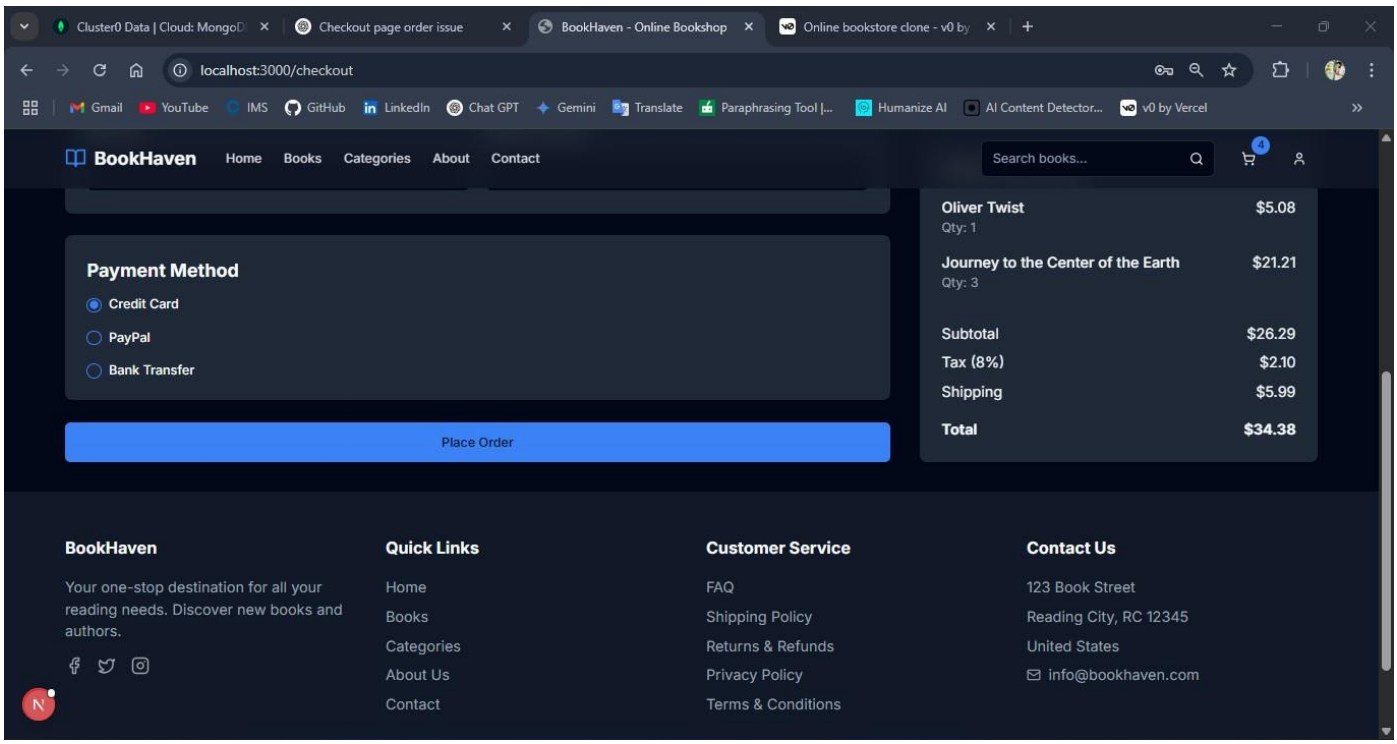
```
export default function CartPage() {
                      <span>Subtotal</span>
                      <span>{formatPrice(subtotal)}</span>
                    </div>
                    <div className="flex justify-between">
                      <span>Tax (8%)</span>
                      <span>{formatPrice(tax)}</span>
                    </div>
                    <div className="flex justify-between">
                      <span>Shipping</span>
                      <span>{formatPrice(shipping)}</span>
                    </div>
                    <div className="border-t pt-2 mt-2">
                      <div className="flex justify-between font-bold">
                        <span>Total</span>
                        <span>{formatPrice(total)}</span>
                      </div>
                    </div>
                  </div>
                  <Button className="w-full mt-6" size="lg" onClick={handleCheckout}>
                    Proceed to Checkout
                  </Button>
                </div>
              </div>
            </div>
          </div>
        )
      }
```

1.4. Checkout Process

- Checkout form screenshot

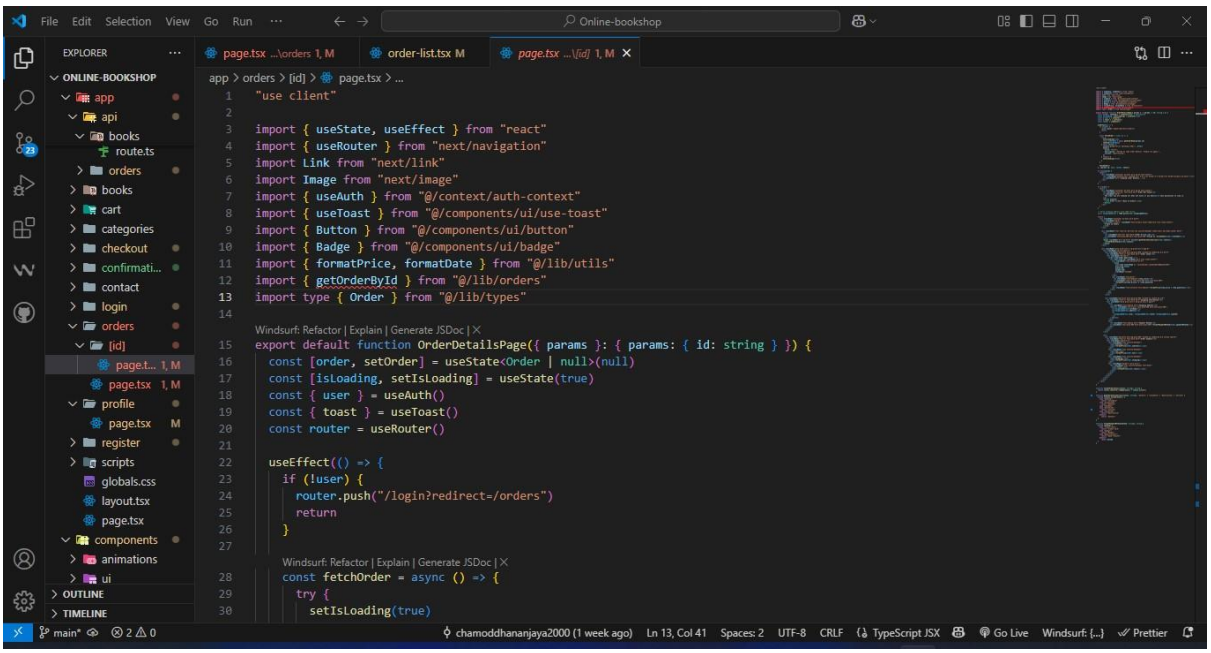


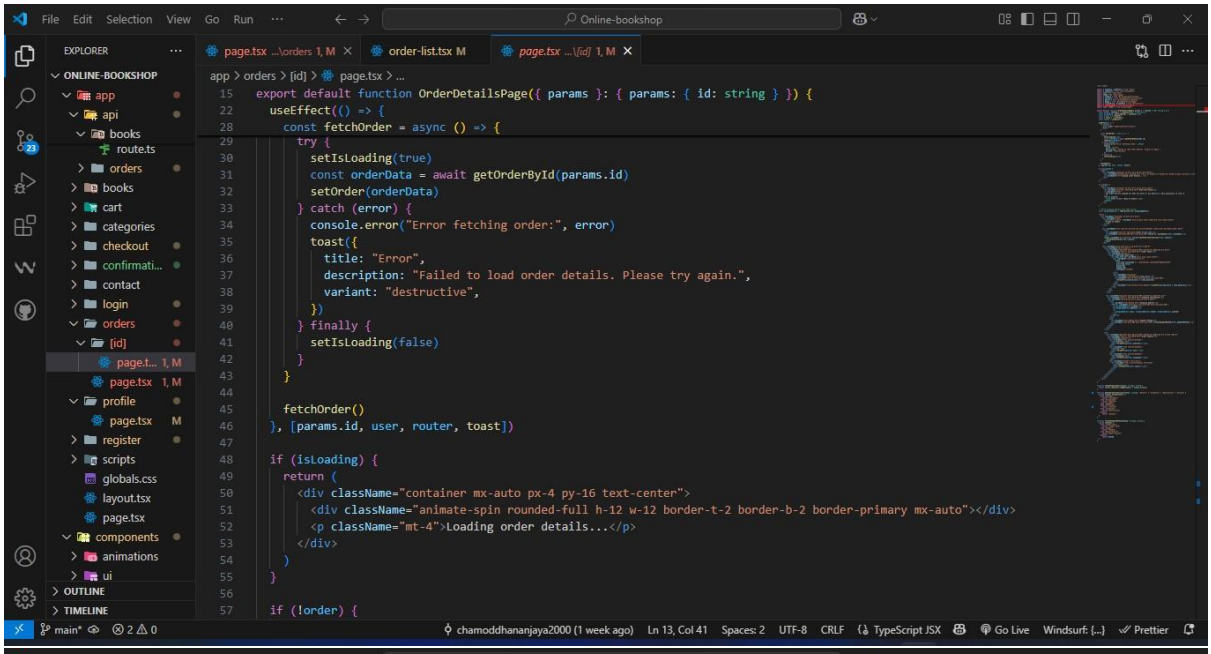
- Payment interface screenshot



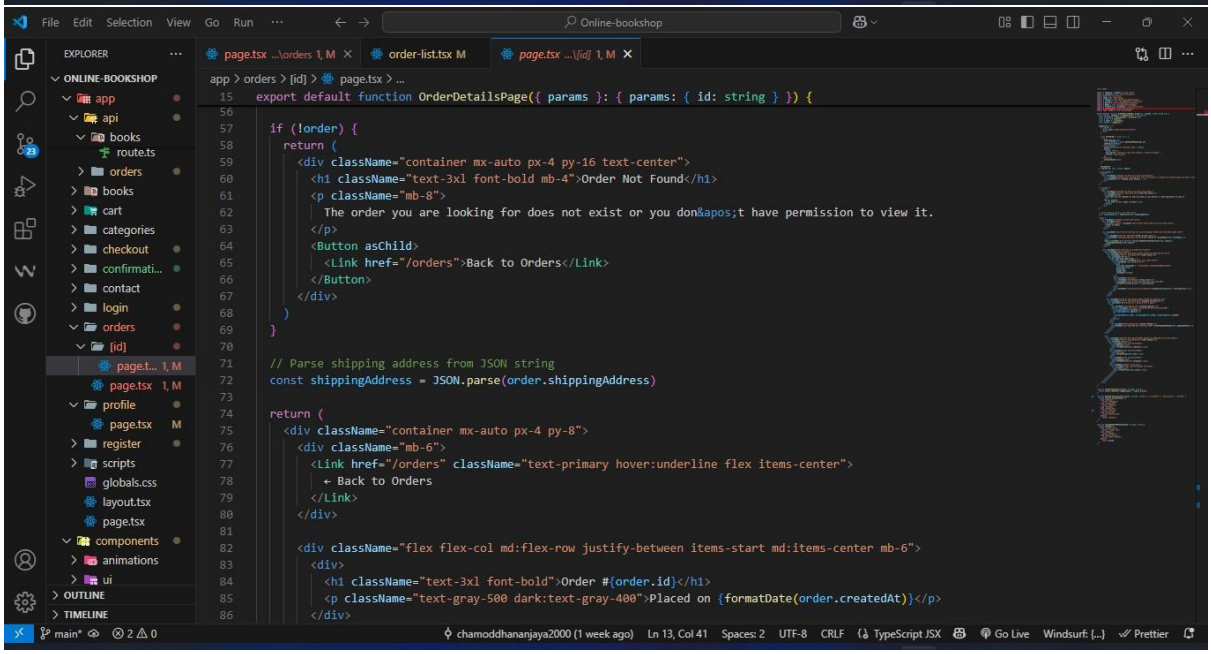
- Order confirmation page screenshot

- Implementation code screenshots

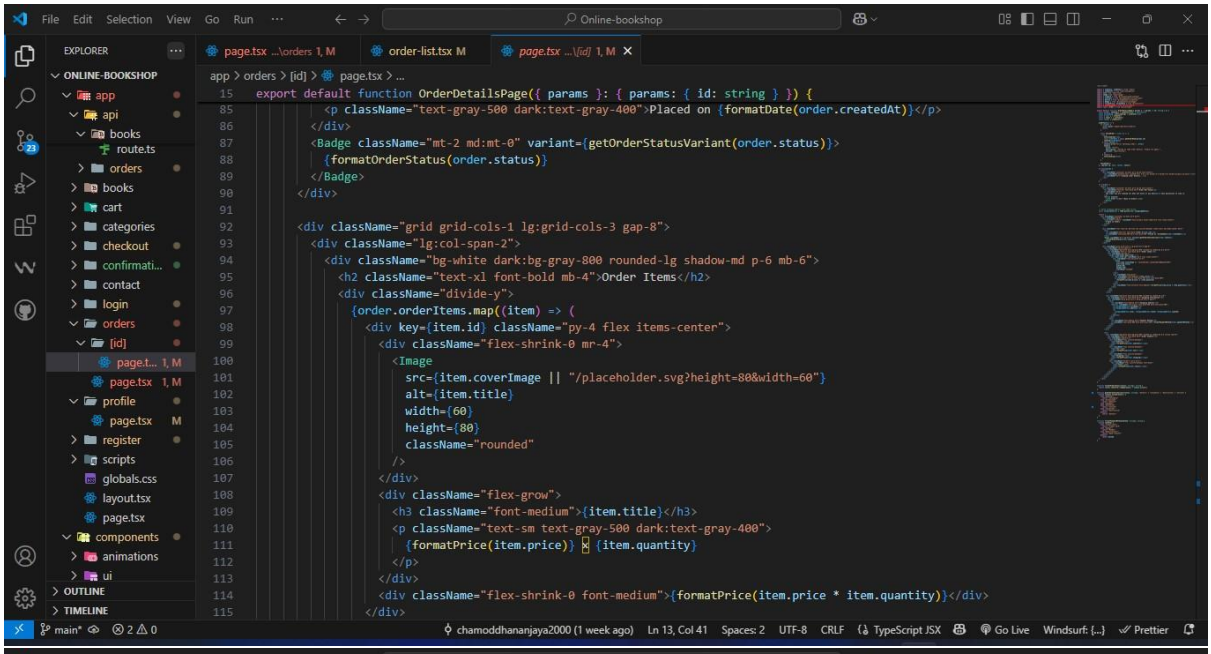




```
export default function OrderDetailsPage({ params }: { params: { id: string } }) {
  useEffect(() => {
    const fetchOrder = async () => {
      try {
        setIsLoading(true)
        const orderData = await getOrderById(params.id)
        setOrder(orderData)
      } catch (error) {
        console.error("Error fetching order:", error)
        toast({
          title: "Error",
          description: "Failed to load order details. Please try again.",
          variant: "destructive",
        })
      } finally {
        setIsLoading(false)
      }
    }

    fetchOrder()
  }, [params.id, user, router, toast])

  if (isLoading) {
    return (
      <div className="container mx-auto px-4 py-16 text-center">
        <div className="animate-spin rounded-full h-12 w-12 border-t-2 border-b-2 border-primary mx-auto"></div>
        <p className="mt-4">Loading order details...</p>
      </div>
    )
  }

  if (!order) {
```



```
export default function OrderDetailsPage({ params }: { params: { id: string } }) {

  if (!order) {
    return (
      <div className="container mx-auto px-4 py-16 text-center">
        <h1 className="text-3xl font-bold mb-4">Order Not Found</h1>
        <p className="mb-8">
          The order you are looking for does not exist or you don&apos;t have permission to view it.
        </p>
        <Button asChild>
          <Link href="/orders">Back to Orders</Link>
        </Button>
      </div>
    )
  }

  // Parse shipping address from JSON string
  const shippingAddress = JSON.parse(order.shippingAddress)

  return (
    <div className="container mx-auto px-4 py-8">
      <div className="mb-6">
        <Link href="/orders" className="text-primary hover:underline flex items-center">
          ← Back to Orders
        </Link>
      </div>

      <div className="flex flex-col md:flex-row justify-between items-start md:items-center mb-6">
        <div>
          <h1 className="text-3xl font-bold">Order #{order.id}</h1>
          <p className="text-gray-500 dark:text-gray-400">Placed on {formatDate(order.createdAt)}</p>
        </div>
```


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx ...\orders 1, M
order-list.tsx M
page.tsx ...\[id] 1, M
app > orders > [id] > page.tsx > ...
export default function OrderDetailsPage({ params }: { params: { id: string } }) {
Placed on {formatDate(order.createdAt)}

<Badge className="mt-2 md:mt-0" variant={getOrderStatusVariant(order.status)}>
{formatOrderStatus(order.status)}
</Badge>

<h2 className="text-xl font-bold mb-4">Order Items</h2>

{order.orderItems.map((item) => (

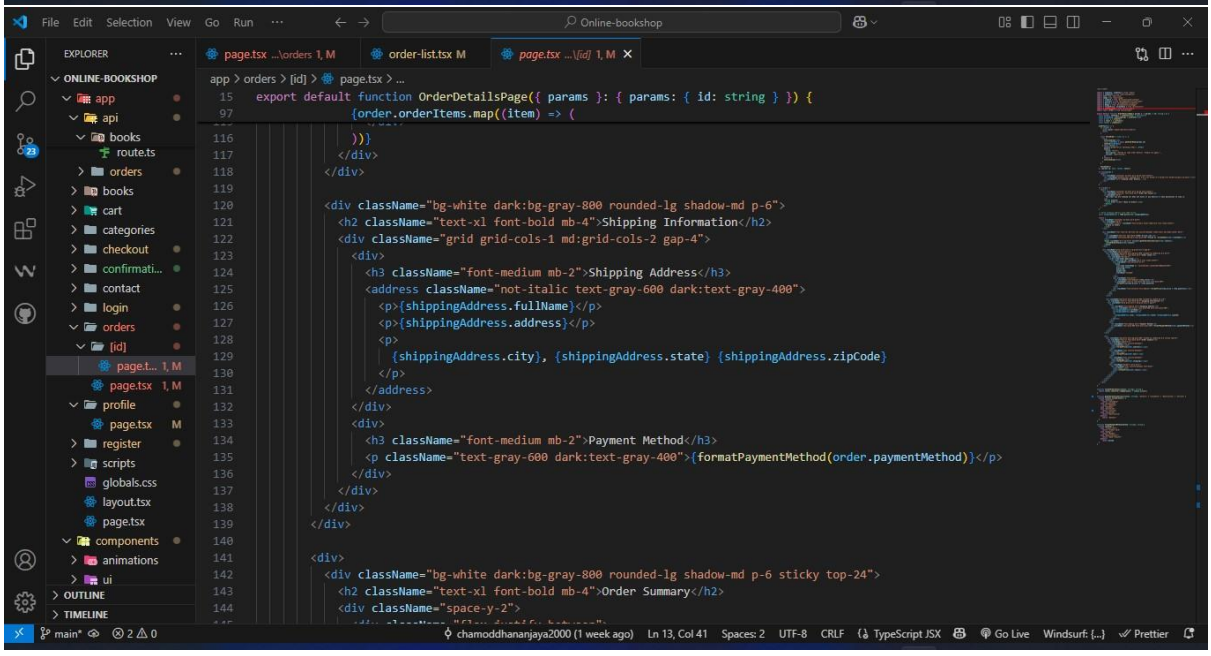
<Image
src={item.coverImage || "/placeholder.svg?height=80&width=60"}
alt={item.title}
width={60}
height={80}
className="rounded"
/>

<h3 className="font-medium">{item.title}</h3>

{formatPrice(item.price)} × {item.quantity}

{formatPrice(item.price * item.quantity)}

chamoddhananjaya2000 (1 week ago) Ln 13, Col 41 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Windsurf: {...} Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
page.tsx ...\orders 1, M
order-list.tsx M
page.tsx ...\[id] 1, M
app > orders > [id] > page.tsx > ...
export default function OrderDetailsPage({ params }: { params: { id: string } }) {
{order.orderItems.map((item) => (
))}

<h2 className="text-xl font-bold mb-4">Shipping Information</h2>

<h3 className="font-medium mb-2">Shipping Address</h3>
<address className="not-italic text-gray-600 dark:text-gray-400">
{shippingAddress.fullName}
{shippingAddress.address}

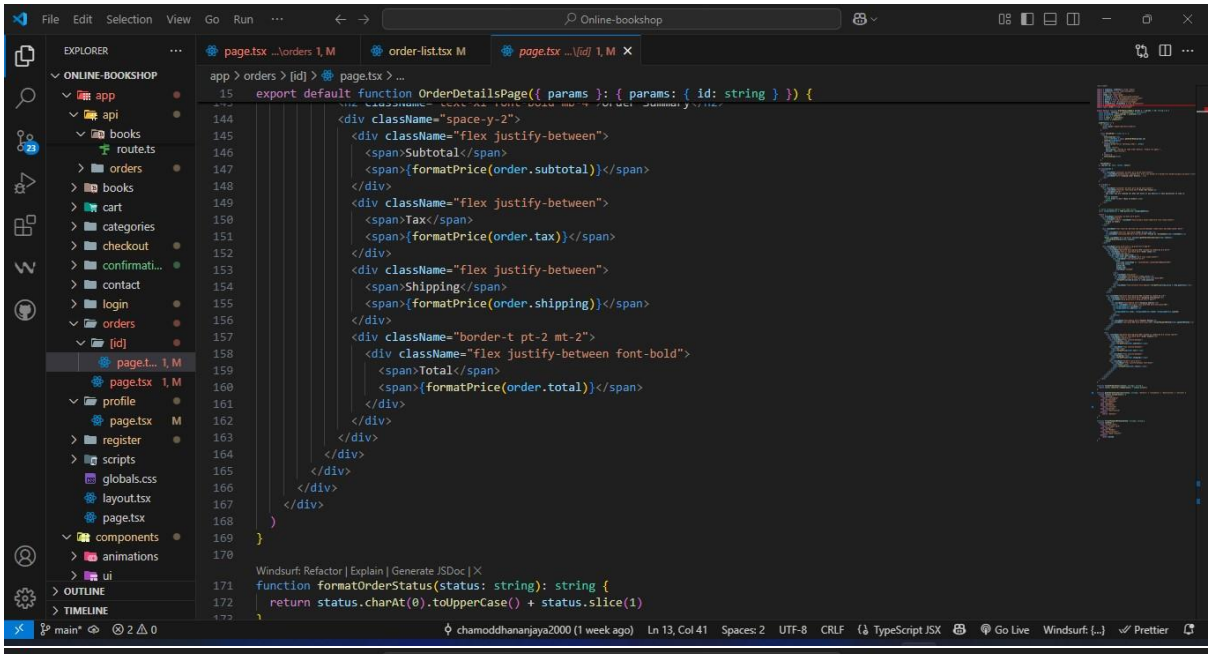
{shippingAddress.city}, {shippingAddress.state} {shippingAddress.zipCode}

</address>

<h3 className="font-medium mb-2">Payment Method</h3>
{formatPaymentMethod(order.paymentMethod)}

<h2 className="text-xl font-bold mb-4">Order Summary</h2>

chamoddhananjaya2000 (1 week ago) Ln 13, Col 41 Spaces: 2 UTF-8 CRLF TypeScript JSX Go Live Windsurf: {...} Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
ONLINE-BOOKSHOP
page.tsx ...\orders 1, M
order-list.tsx M
page.tsx ...\[id] 1, M
app > orders > [id] > page.tsx > ...
export default function OrderDetailsPage({ params }: { params: { id: string } }) {

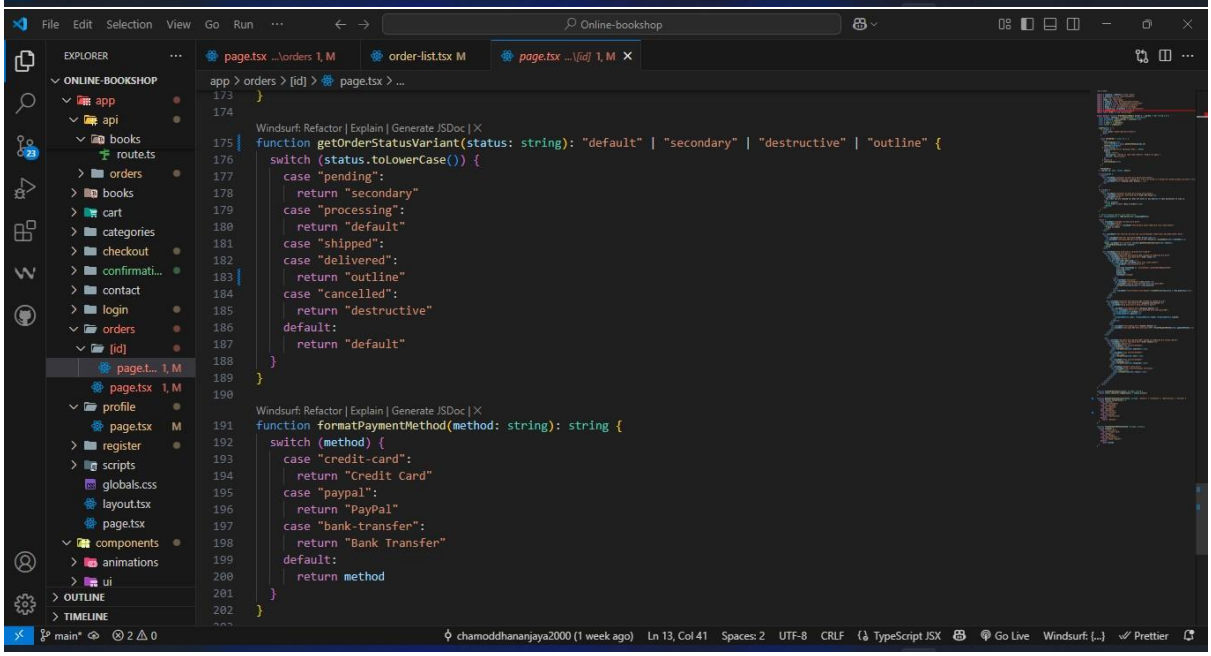
Subtotal
{formatPrice(order.subtotal)}

Tax
{formatPrice(order.tax)}

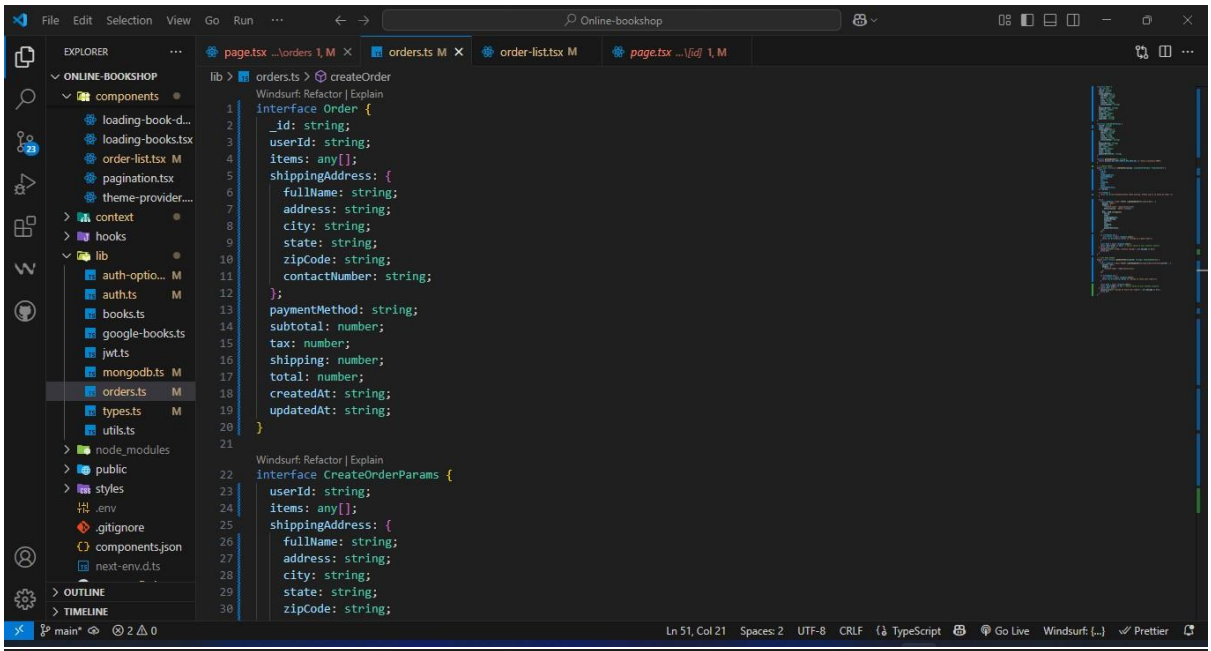
Shipping
{formatPrice(order.shipping)}

Total
{formatPrice(order.total)}

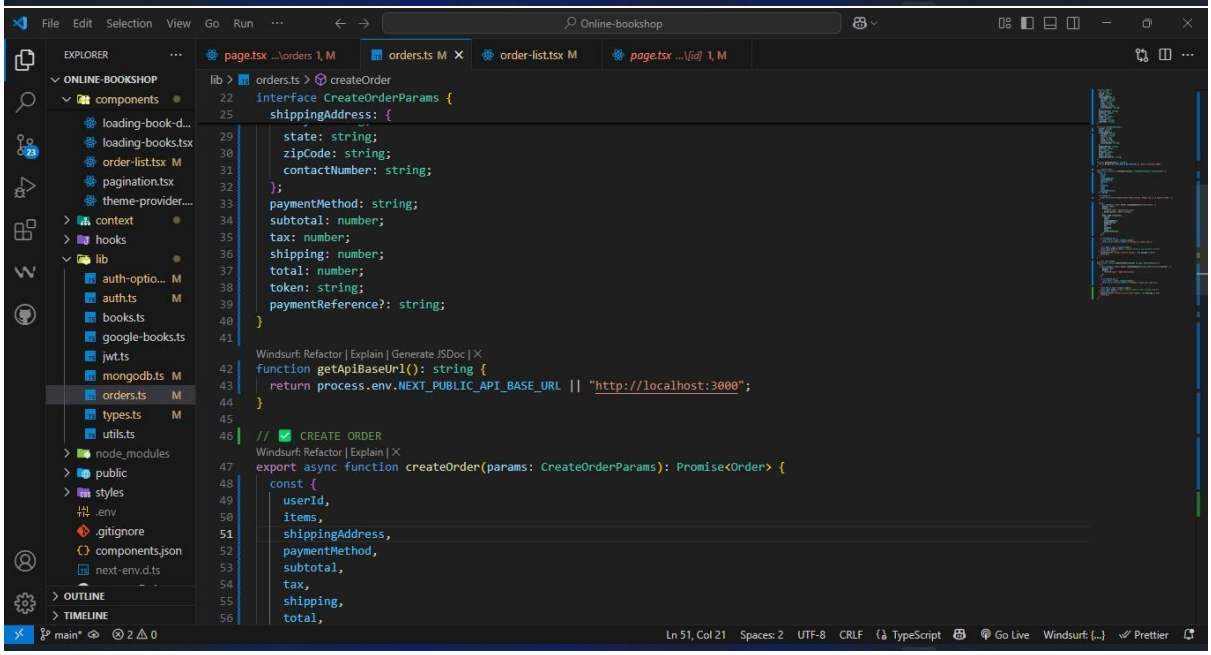
)
}
Windsurf: Refactor | Explain | Generate JSDoc |
function formatOrderStatus(status: string): string {
return status.charAt(0).toUpperCase() + status.slice(1)
OUTLINE
TIMELINE
main*
chamoddhananjaya2000 (1 week ago)
Ln 13, Col 41
Spaces: 2
UTF-8
CRLF
TypeScript JSX
Go Live
Windsurf: {...}
Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
ONLINE-BOOKSHOP
page.tsx ...\orders 1, M
order-list.tsx M
page.tsx ...\[id] 1, M
app > orders > [id] > page.tsx > ...
}
Windsurf: Refactor | Explain | Generate JSDoc |
function getOrderStatusVariant(status: string): "default" | "secondary" | "destructive" | "outline" {
switch (status.toLowerCase()) {
case "pending":
return "secondary"
case "processing":
return "default"
case "shipped":
case "delivered":
return "outline"
case "cancelled":
return "destructive"
default:
return "default"
}
}
Windsurf: Refactor | Explain | Generate JSDoc |
function formatPaymentMethod(method: string): string {
switch (method) {
case "credit-card":
return "Credit Card"
case "paypal":
return "PayPal"
case "bank-transfer":
return "Bank Transfer"
default:
return method
}
}
OUTLINE
TIMELINE
main*
chamoddhananjaya2000 (1 week ago)
Ln 13, Col 41
Spaces: 2
UTF-8
CRLF
TypeScript JSX
Go Live
Windsurf: {...}
Prettier

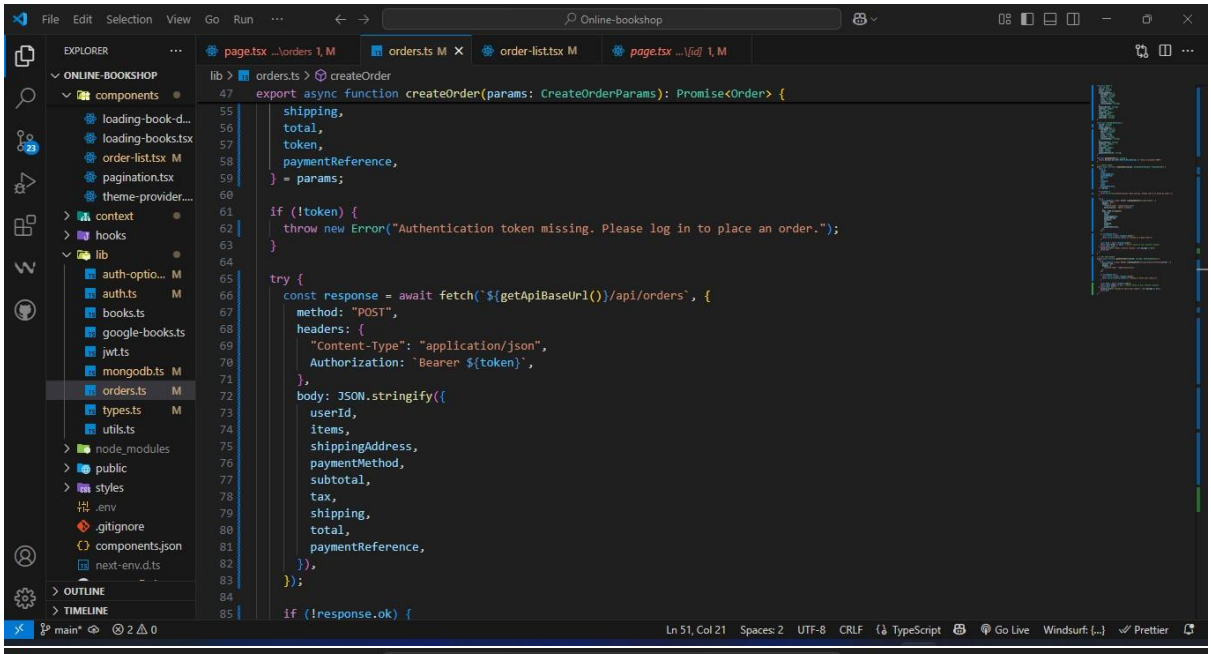
- Order processing code screenshots

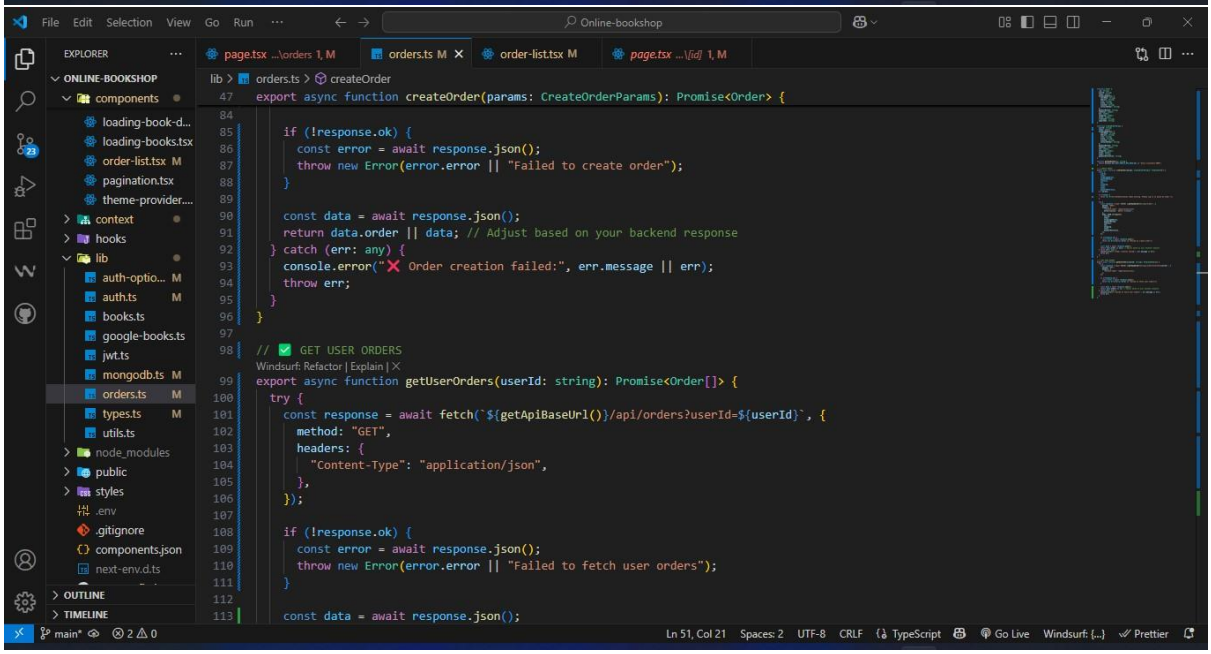


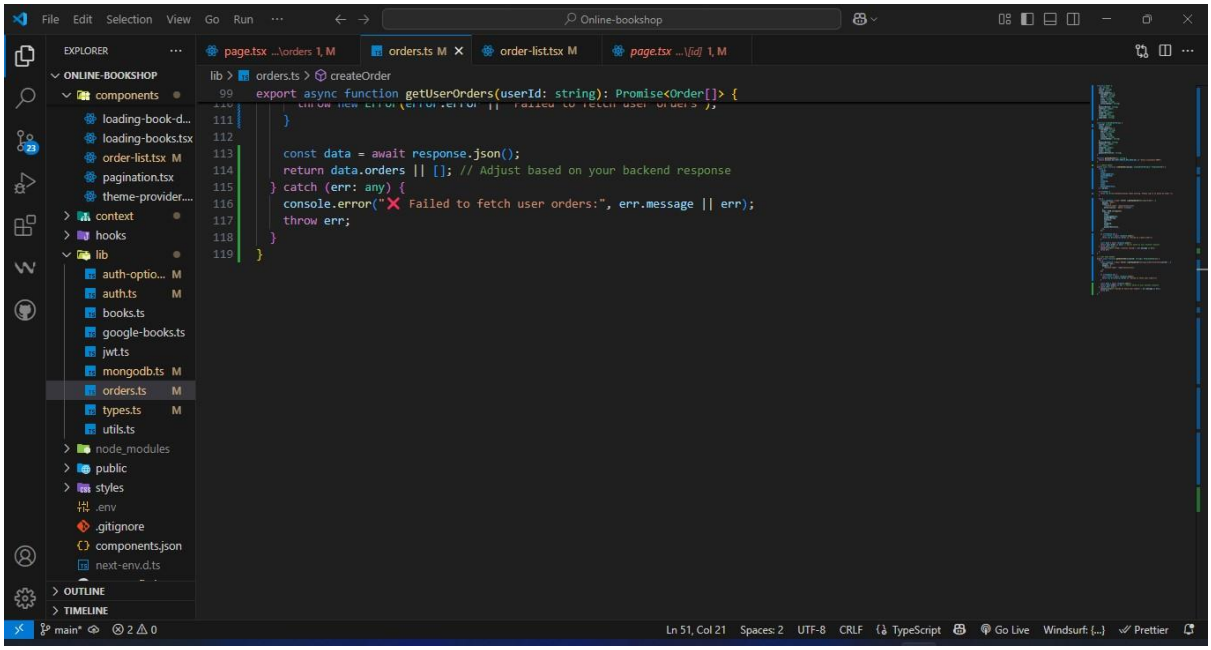
```
interface Order {
  _id: string;
  userId: string;
  items: any[];
  shippingAddress: {
    fullName: string;
    address: string;
    city: string;
    state: string;
    zipCode: string;
    contactNumber: string;
  };
  paymentMethod: string;
  subtotal: number;
  tax: number;
  shipping: number;
  total: number;
  createdAt: string;
  updatedAt: string;
}

interface CreateOrderParams {
  userId: string;
  items: any[];
  shippingAddress: {
    fullName: string;
    address: string;
    city: string;
    state: string;
    zipCode: string;
```



```
interface CreateOrderParams {
  shippingAddress: {
    state: string;
    zipCode: string;
    contactNumber: string;
  };
  paymentMethod: string;
  subtotal: number;
  tax: number;
  shipping: number;
  total: number;
  token: string;
  paymentReference?: string;
}

function getApiBaseUrl(): string {
  return process.env.NEXT_PUBLIC_API_BASE_URL || "http://localhost:3000";
}

// ✅ CREATE ORDER
export async function createOrder(params: CreateOrderParams): Promise<Order> {
  const {
    userId,
    items,
    shippingAddress,
    paymentMethod,
    subtotal,
    tax,
    shipping,
    total,
```

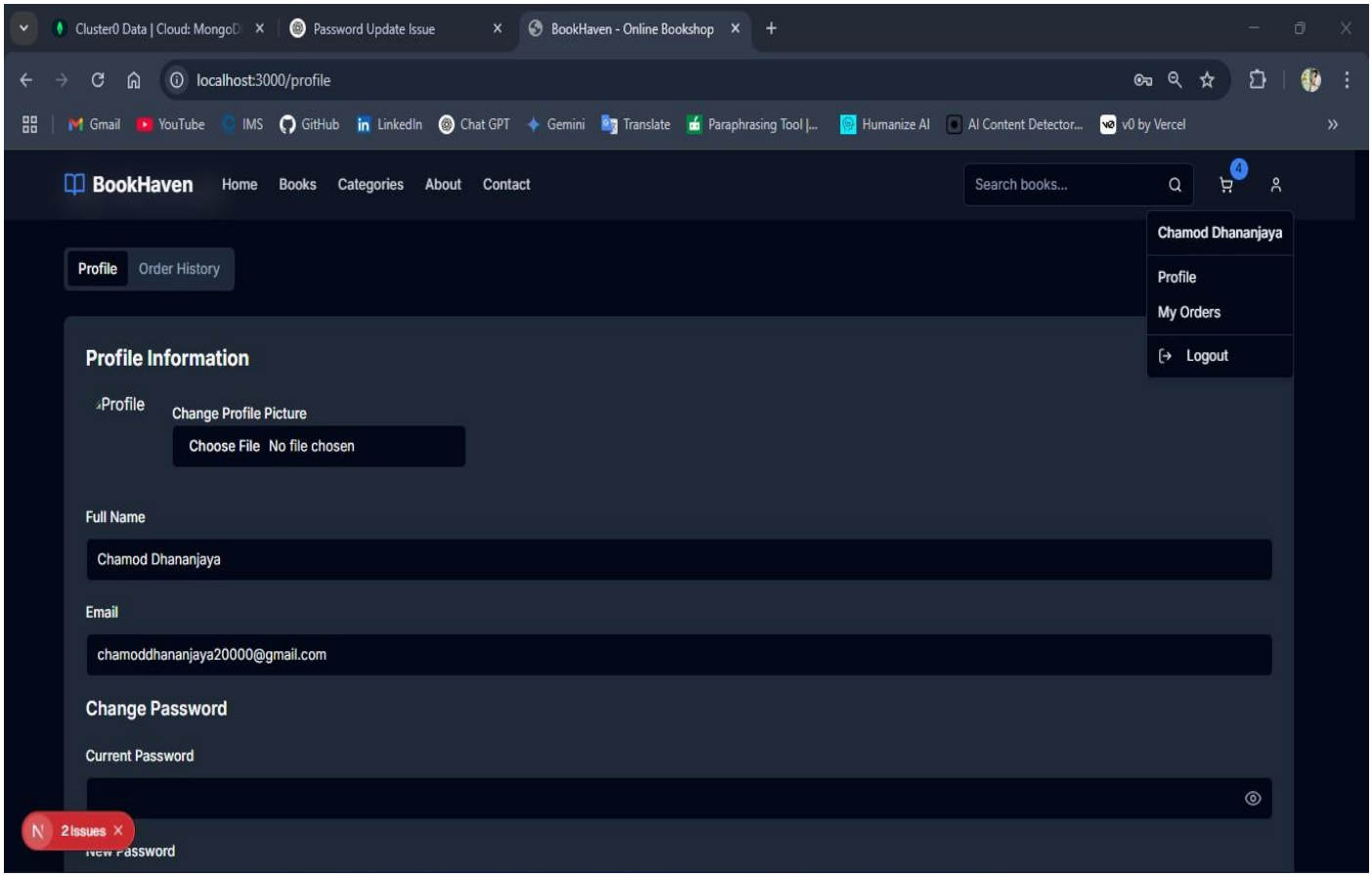

File Edit Selection View Go Run
Online-bookshop
EXPLORER
ONLINE-BOOKSHOP
components
loading-book-d...
loading-books.tsx
order-list.tsx M
pagination.tsx
theme-provider....
context
hooks
lib
auth-optio... M
auth.ts M
books.ts
google-books.ts
jwt.ts
mongodb.ts M
orders.ts M
types.ts M
utils.ts
node_modules
public
styles
.env
.gitignore
components.json
next-env.d.ts
OUTLINE
TIMELINE
page.tsx ...\orders 1, M
orders.ts M
order-list.tsx M
page.tsx ...\[id] 1, M
lib > orders.ts > createOrder
export async function createOrder(params: CreateOrderParams): Promise<Order> {
shipping,
total,
token,
paymentReference,
} = params;
if (!token) {
throw new Error("Authentication token missing. Please log in to place an order.");
}
try {
const response = await fetch(`${getApiBaseUrl()}/api/orders`, {
method: "POST",
headers: {
"Content-Type": "application/json",
Authorization: `Bearer ${token}`,
},
body: JSON.stringify({
userId,
items,
shippingAddress,
paymentMethod,
subtotal,
tax,
shipping,
total,
paymentReference,
}),
});
if (!response.ok) {
main* 2 0
Ln 51, Col 21 Spaces: 2 UTF-8 CRLF TypeScript Go Live Windsurf: {...} Prettier


File Edit Selection View Go Run
Online-bookshop
EXPLORER
ONLINE-BOOKSHOP
components
loading-book-d...
loading-books.tsx
order-list.tsx M
pagination.tsx
theme-provider....
context
hooks
lib
auth-optio... M
auth.ts M
books.ts
google-books.ts
jwt.ts
mongodb.ts M
orders.ts M
types.ts M
utils.ts
node_modules
public
styles
.env
.gitignore
components.json
next-env.d.ts
OUTLINE
TIMELINE
page.tsx ...\orders 1, M
orders.ts M
order-list.tsx M
page.tsx ...\[id] 1, M
lib > orders.ts > createOrder
export async function createOrder(params: CreateOrderParams): Promise<Order> {
if (!response.ok) {
const error = await response.json();
throw new Error(error.error || "Failed to create order");
}
const data = await response.json();
return data.order || data; // Adjust based on your backend response
} catch (err: any) {
console.error("❌ Order creation failed:", err.message || err);
throw err;
}
}
// ✅ GET USER ORDERS
Windsurf: Refactor | Explain |
export async function getUserOrders(userId: string): Promise<Order[]> {
try {
const response = await fetch(`${getApiBaseUrl()}/api/orders?userId=${userId}`, {
method: "GET",
headers: {
"Content-Type": "application/json",
},
});
if (!response.ok) {
const error = await response.json();
throw new Error(error.error || "Failed to fetch user orders");
}
const data = await response.json();
main* 2 0
Ln 51, Col 21 Spaces: 2 UTF-8 CRLF TypeScript Go Live Windsurf: {...} Prettier

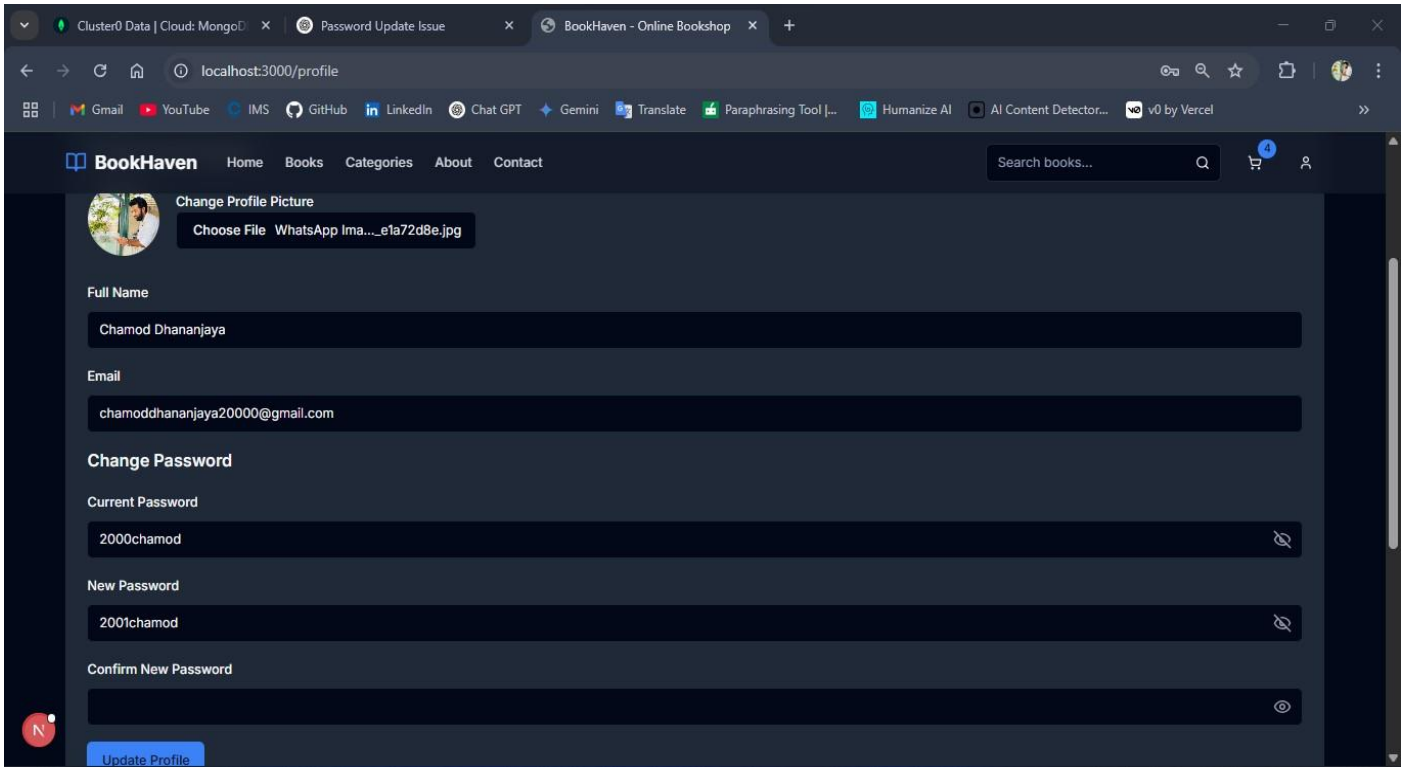

File Edit Selection View Go Run
Online-bookshop
EXPLORER
ONLINE-BOOKSHOP
components
loading-book-d...
loading-books.tsx
order-list.tsx M
pagination.tsx
theme-provider....
context
hooks
lib
auth-optio... M
auth.ts M
books.ts
google-books.ts
jwt.ts
mongodb.ts M
orders.ts M
types.ts M
utils.ts
node_modules
public
styles
.env
.gitignore
components.json
next-env.d.ts
OUTLINE
TIMELINE
page.tsx ...\orders 1, M
orders.ts M
order-list.tsx M
page.tsx ...\[id] 1, M
lib > orders.ts > createOrder
99 export async function getUserOrders(userId: string): Promise<Order[]> {
111 }
112
113 const data = await response.json();
114 return data.orders || []; // Adjust based on your backend response
115 } catch (err: any) {
116 console.error("❌ Failed to fetch user orders:", err.message || err);
117 throw err;
118 }
119 }
main* 2 0
Ln 51, Col 21 Spaces: 2 UTF-8 CRLF TypeScript Go Live Windsurf: {...} Prettier

1.5. User Profile

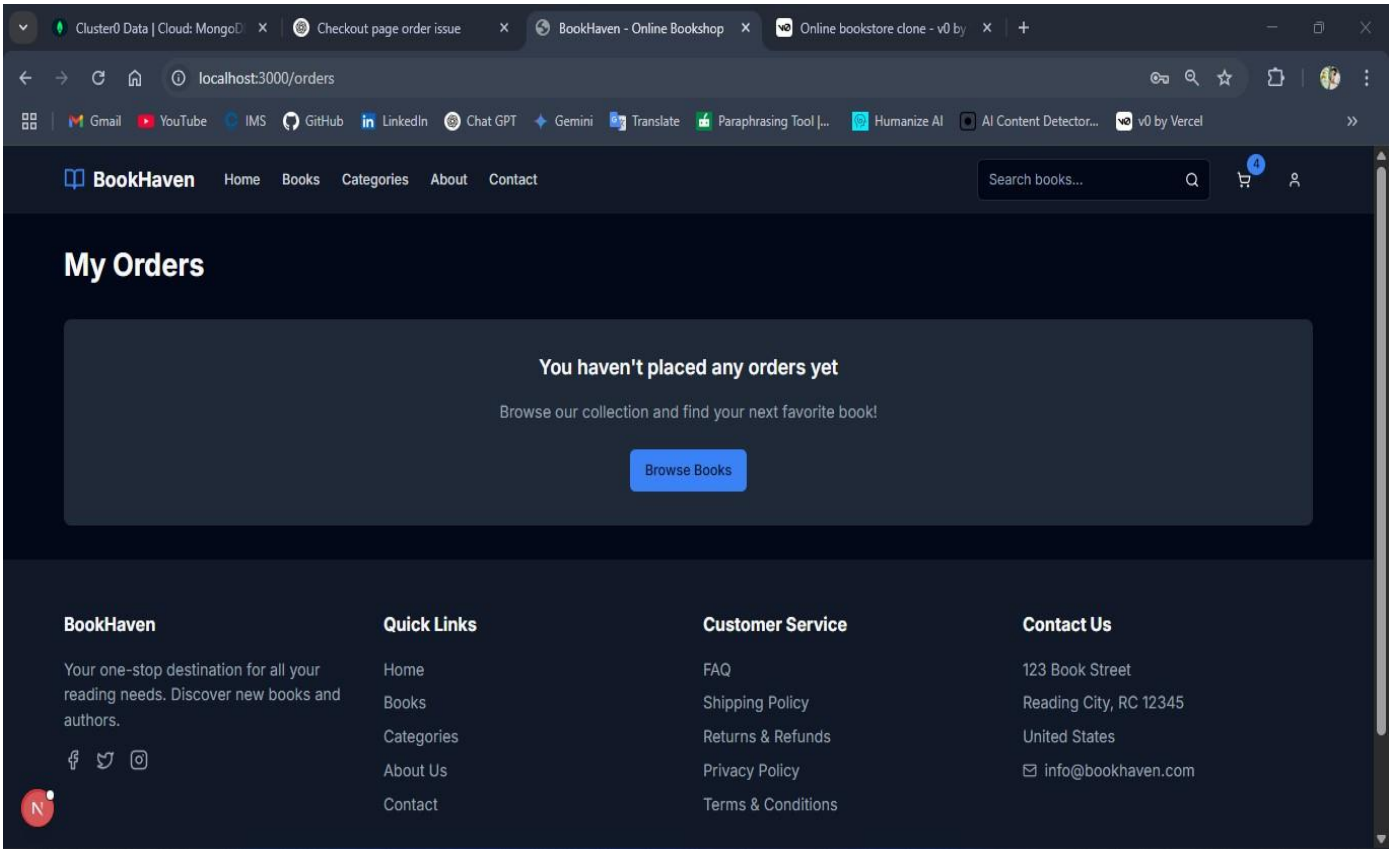
- Profile view screenshot



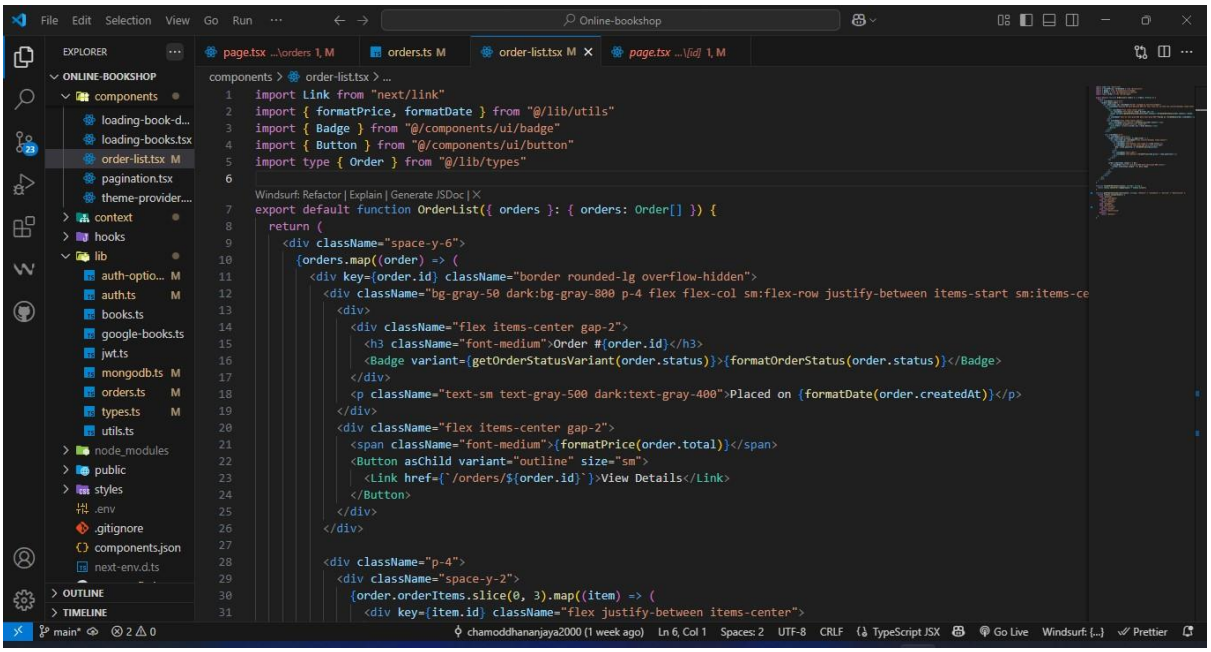
- Edit profile interface screenshot

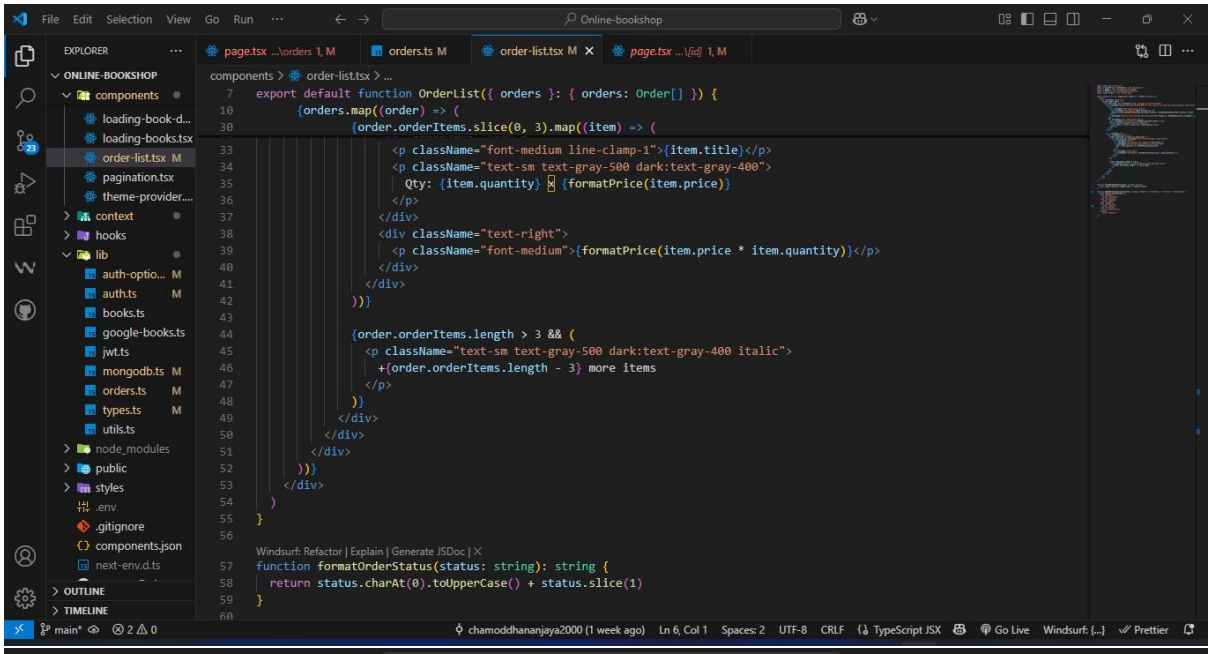


- Order history view screenshot

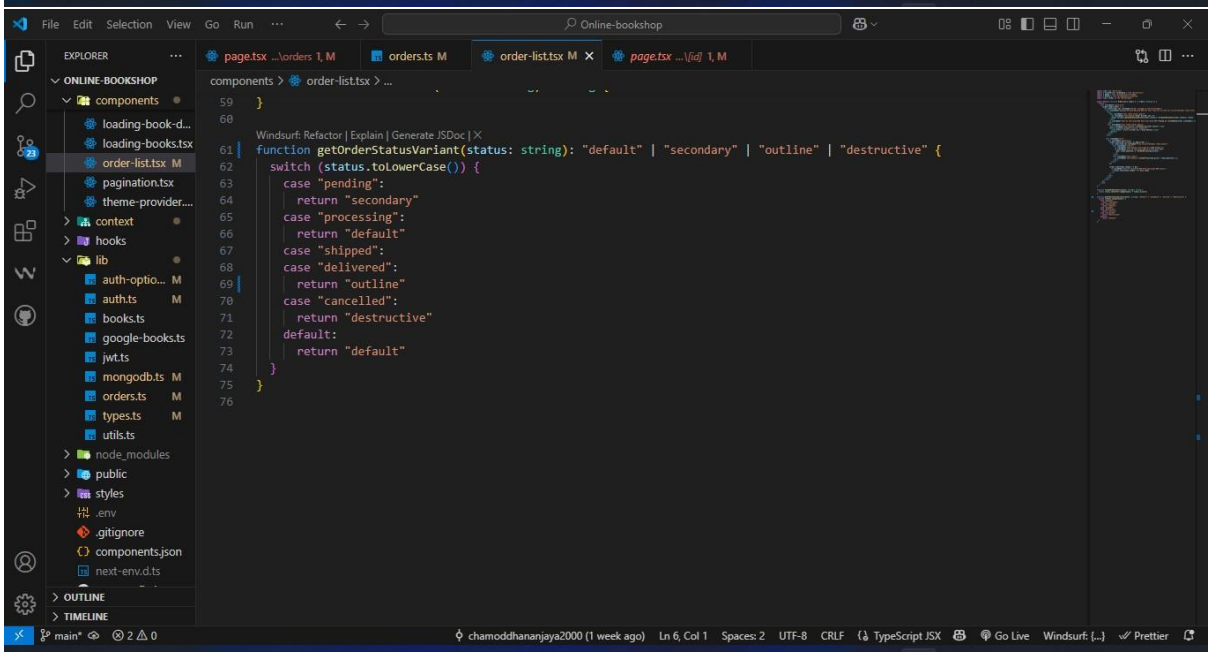


- Related implementation code screenshots





```
export default function OrderList({ orders }: { orders: Order[] }) {
      {orders.map((order) => (
              {order.orderItems.slice(0, 3).map((item) => (
                      <p className="font-medium line-clamp-1">{item.title}</p>
                      <p className="text-sm text-gray-500 dark:text-gray-400">
                        Qty: {item.quantity} x {formatPrice(item.price)}
                      </p>
                    </div>
                    <div className="text-right">
                      <p className="font-medium">{formatPrice(item.price * item.quantity)}</p>
                    </div>
                  </div>
                ))}

                {order.orderItems.length > 3 && (
                  <p className="text-sm text-gray-500 dark:text-gray-400 italic">
                    +{order.orderItems.length - 3} more items
                  </p>
                )}
              </div>
            </div>
          </div>
        ))}
      </div>
    )
}

Windsurf: Refactor | Explain | Generate JSDoc | X
function formatOrderStatus(status: string): string {
  return status.charAt(0).toUpperCase() + status.slice(1)
}
```



```
}

Windsurf: Refactor | Explain | Generate JSDoc | X
function getOrderStatusVariant(status: string): "default" | "secondary" | "outline" | "destructive" {
  switch (status.toLowerCase()) {
    case "pending":
      return "secondary"
    case "processing":
      return "default"
    case "shipped":
    case "delivered":
      return "outline"
    case "cancelled":
      return "destructive"
    default:
      return "default"
  }
}
```