

**Smart Trial Studio**  
**Kashmira Chaudhari 20158**

## 1. Introduction

1. Revolutionizing the way we shop for clothing: The Smart Trial Studio offers an innovative approach to shopping. Customers can effortlessly curate a trial cart with up to 5 items from various brands and styles, simplifying the selection process.
2. Seamless Integration with Retailers: Leveraging a network of online and offline retailers, customers can conveniently try on their chosen items at a nearby store's trial room within a 1-day window. This bridges the gap between online browsing and in-store trying.
3. Enhanced Shopping Experience: This service reduces the time and hassle of traditional shopping while enhancing the overall experience with personalized convenience and flexibility.
4. API Development Goals:
  - CRUD Operations: My goal is to develop API's that can perform different operations like Create, Read, Update, and Delete (CRUD) on the database.
  - Access Control:
    - READ Access: Available to both employees and customers.
    - CREATE, UPDATE, and DELETE Access: Restricted to store employees only.

## 2. Tech Stack:

- I. **Springboot:** Java Spring Boot is an open-source tool that makes it easier to use Java-based frameworks to create microservices and web apps.
- II. **Mongodb:** MongoDB is the document database designed to make it easy for developers to work with data in any form, and from any programming language.
- III. **Java:** Java is a versatile, object-oriented programming language designed for building platform-independent, secure, and high-performance applications.
- IV. **HTML:** HTML (HyperText Markup Language) is the standard language used to create and structure content on the web by defining elements like text, images, and links.
- V. **JavaScript:** JavaScript is a dynamic, high-level programming language primarily used to create interactive and responsive behavior on web pages.
- VI. **CSS:** CSS (Cascading Style Sheets) is a stylesheet language used to control the presentation, layout, and design of HTML elements on a web page.

### 3. Architecture Diagram:

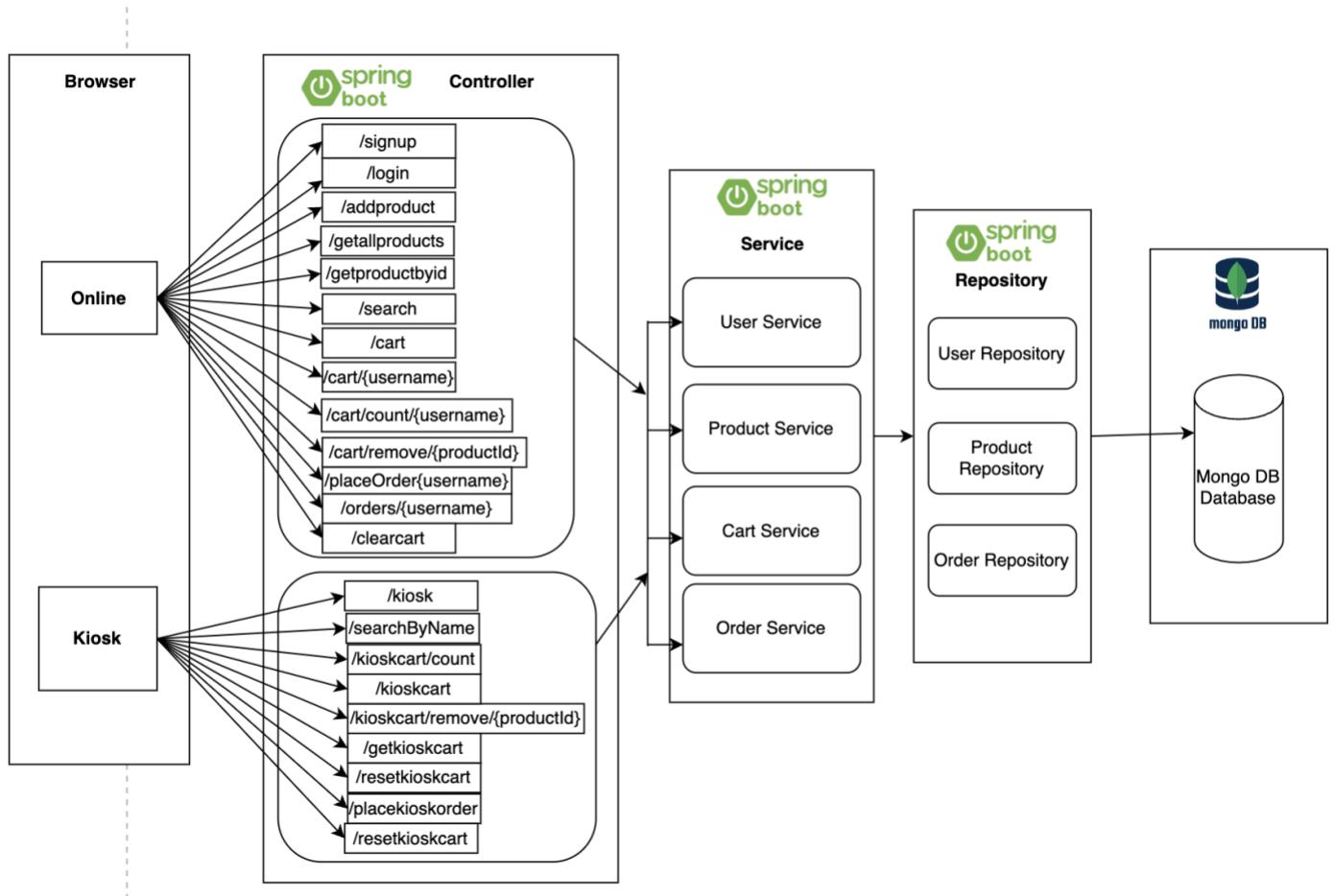


Fig: Architecture of Smart Trial Studio

## 4. Use Case for Smart Trial Studio

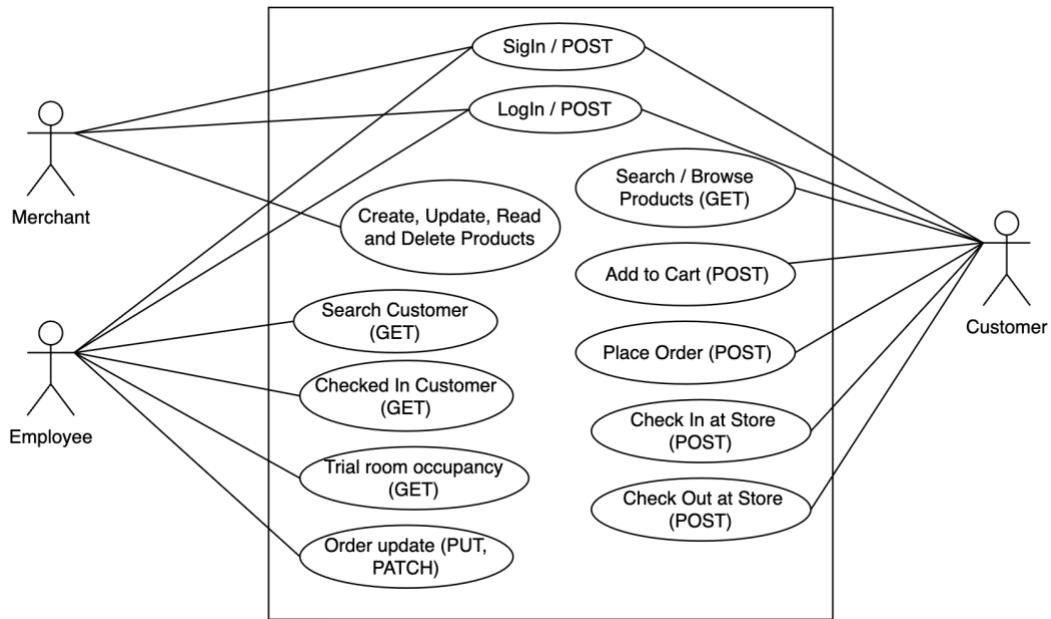


Fig: Smart Trial Studio

## 5. FLOW chart:

### I. User/Customer Flowchart

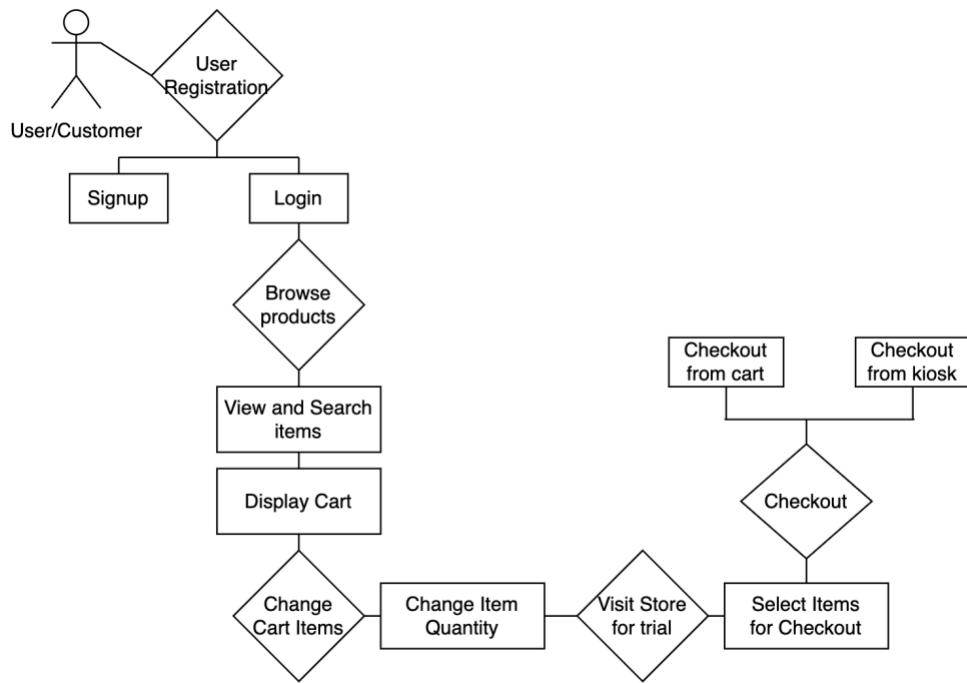


Fig: Flowchart for Customer

## II. Store Flowchart:

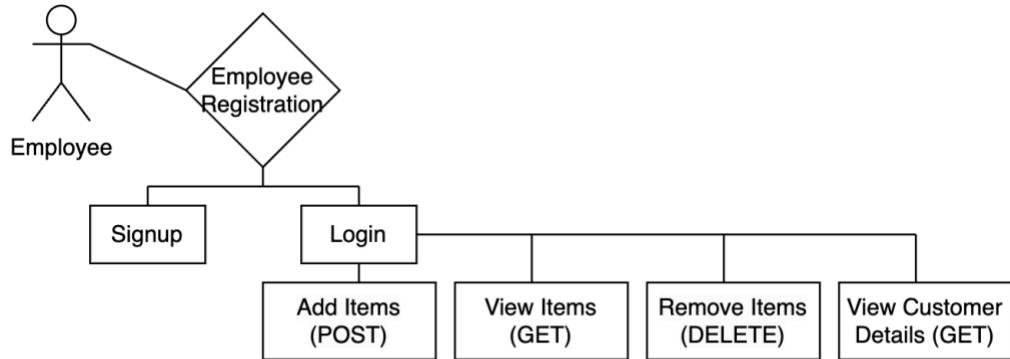


Fig: Flowchart for Store Employee

## 6. Setting up MongoDB:

← → ⌛ cloud.mongodb.com/v2/6688800d910c042ef26d844c#/clusters/starterTemplates

Deploy your cluster

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

<input type="radio"/> <b>M10</b>	<b>\$0.08/hour</b>	
For production applications with sophisticated workload requirements.		
STORAGE 10 GB	RAM 2 GB	vCPU 2 vCPUs

<input type="radio"/> <b>Serverless</b>	<b>\$0.10/1M reads</b>	
For application development and testing, or workloads with variable traffic.		
STORAGE Up to 1TB	RAM Auto-scale	vCPU Auto-scale

<input checked="" type="radio"/> <b>M0</b>	<b>Free</b>	
For learning and exploring MongoDB in a cloud environment.		
STORAGE 512 MB	RAM Shared	vCPU Shared

 **Free forever!** Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Name  
You cannot change the name once the cluster is created.

Automate security setup i  
 Preload sample dataset i



← → ⌛ cloud.mongodb.com/v2/6688800d910c042ef26d844c#/clusters/starterTemplates

 **Free forever!** Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Name  
You cannot change the name once the cluster is created.

Automate security setup i  
 Preload sample dataset i

Provider  
  

Region  
 **N. Virginia (us-east-1)**  

★ Recommended i  Low carbon emissions i

Tag (optional)  
Create your first tag to categorize and label your resources; more tags can be added later. [Learn more](#).  
 :

The screenshot shows the MongoDB Atlas Overview page for Project 0. On the left, a sidebar lists various service categories like Database, Data Lake, and Device & Edge Sync. The main area features a large button labeled 'Create a cluster' with a green cylinder icon. Below it, a sub-section titled 'Create a cluster' asks to 'Choose your cloud provider, region, and specs.' A prominent green 'Create' button is at the bottom. To the right, a 'Toolbar' section contains links for 'GENERAL' such as 'Get Started with Atlas', 'Reference MongoDB Documentation', 'Develop Applications with the Developer Center', and 'Ask the MongoDB Community'. Below this is a 'New On Atlas' section with a '6 NEW' badge, followed by a 'Support' section with a 'Basic Plan' link.

## 7. Created a database named “trialStudio”

This screenshot shows the same MongoDB Atlas Overview page, but now with a database named 'trialStudio' selected in the 'Clusters' section. The 'Clusters' panel displays the database name, a success message about a sample dataset being loaded, and buttons for 'Connect', 'Edit configuration', 'Browse collections', and 'View monitoring'. The 'Data Size' is listed as 79.97 MB. Below this is an 'Application Development' section with a 'Select' dropdown and a 'Get connection string' button. The right side of the screen remains largely the same, showing the 'Tasks' panel with a 'Learn How to use Atlas with Mflix Sample Data' guide, the 'Toolbar' with a progress bar, and the 'Support' section.

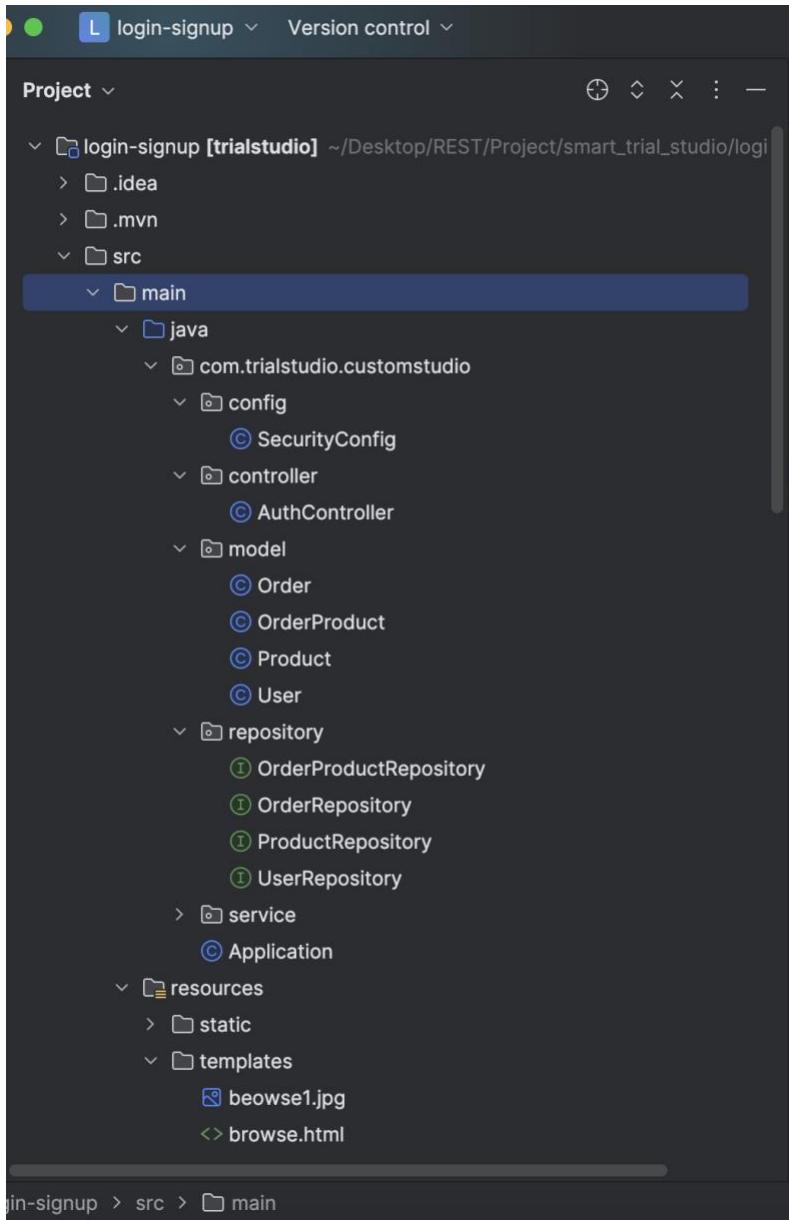
The screenshot shows the MongoDB Atlas interface for the trialStudio database. The left sidebar includes sections for Project 0, Data Services, App Services, and Charts. Under DEPLOYMENT, the Database section shows two databases and four collections. The Collections tab is selected, displaying the trialStudio database with three collections: loginSignin, product, and users. Each collection's details are shown in a table.

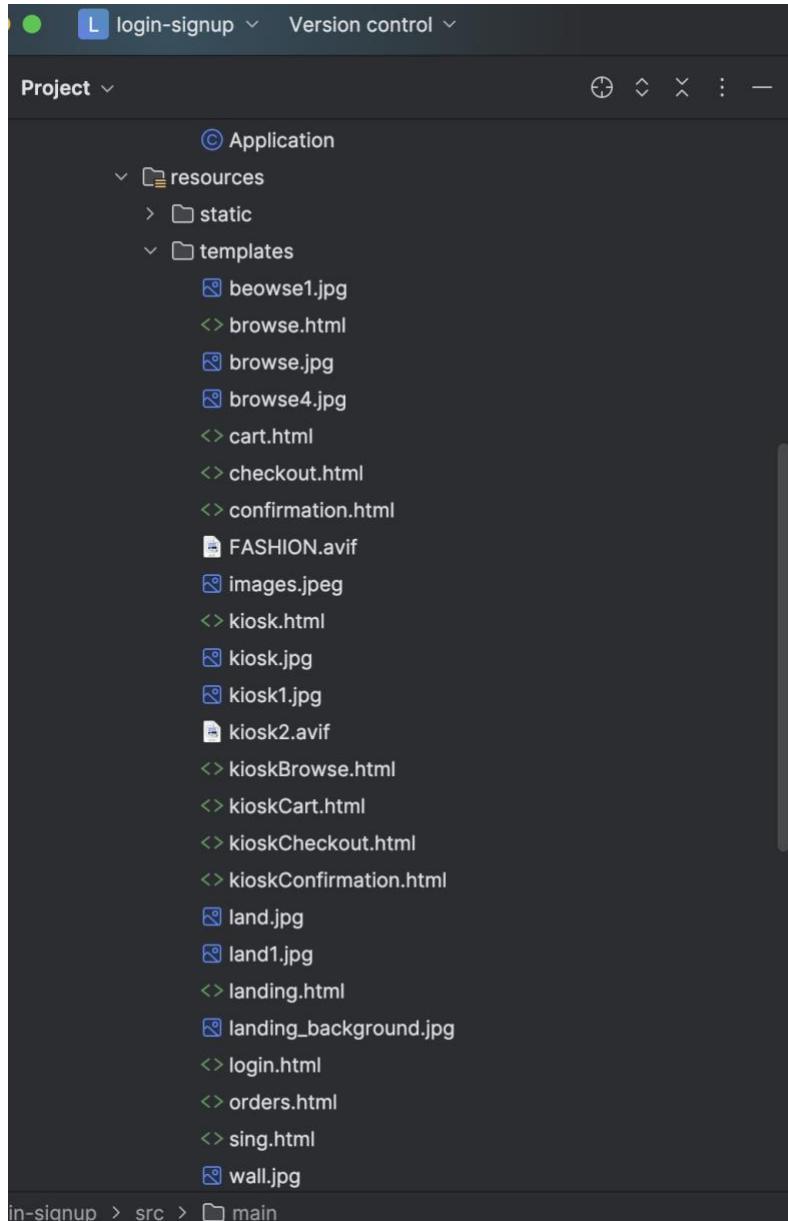
Collection Name	Documents	Logical Data Size	Avg Document Size	Storage Size	Indexes	Index Size	Avg Index Size
loginSignin	0	0B	0B	4KB	1	4KB	4KB
product	3	944B	315B	36KB	1	36KB	36KB
users	14	9.23KB	676B	36KB	1	36KB	36KB

## 8. Create a project using <https://start.spring.io/>

The screenshot shows the start.spring.io interface for creating a new Spring Boot project. The left sidebar has a menu icon. The main area is divided into sections: Project, Language, Spring Boot, Project Metadata, Dependencies, and a preview window. In the Project section, 'Gradle - Groovy' is selected. In the Language section, 'Java' is selected. Under Spring Boot, '3.3.1' is selected. Project Metadata includes fields for Group (com.trialStudio), Artifact (trialStudio), Name (trialStudio), Description (Demo project for Spring Boot), Package name (com.trialStudio.trialStudio), and Packaging (Jar). In the Dependencies section, 'Spring Web' (WEB) and 'Spring Data MongoDB' (NOSQL) are selected. The preview window shows a running Spring Boot application with a dashboard.

## **9. Project Structure:**





## 10. Created below listed API End points:

1. @PostMapping("/signup") => api/auth/signup
2. @PostMapping("/login") => api/auth/login
3. @PostMapping("/addproduct") => /api/auth/addproduct
4. @GetMapping("/getallproducts") => /api/auth/getallproducts
5. @GetMapping("/getproductbyid") => /api/auth/getproductbyid
6. @DeleteMapping("/deleteproductbyid") => /api/auth/deleteproductbyid

7. @GetMapping("/search") => /api/auth/search
8. @PutMapping("/{id}") => /api/auth/{id}
9. @PatchMapping("/{id}") => /api/auth/{id}
10. @GetMapping("/login")
11. @GetMapping("/signup")
12. @PostMapping("/cart")
13. @GetMapping("/cart/{username}")
14. @GetMapping("/cart/count/{username}")
15. @GetMapping("/kiosk/count")
16. @GetMapping("/searchByName")
17. @GetMapping("/searchByName")
18. @PostMapping("/kioskcarts")
19. @DeleteMapping("/kioskcarts/remove/{productId}")
20. @GetMapping("/getkioskcarts")
21. @GetMapping("/kioskcarts/count")
22. @GetMapping("/resetkioskcarts")
23. @PostMapping("/placeOrder/{username}")
24. @GetMapping("orders/{username}")
25. @PostMapping("/placekioskorder")
26. @PostMapping("/clearcart")
27. @PostMapping("/resetkioskcarts")

## 11. AuthController code to write API's:

The screenshot shows a Java code editor with two tabs open:

- AuthController.java**: This tab contains the implementation for the `AuthController`. It includes methods for user registration and login, both returning `ResponseEntity` objects. The `registerUser` method sets the password and saves the user. The `authenticateUser` method checks if the provided credentials match a user in the database and returns a success or failure response.
- CorsConfig.java**: This tab contains configuration code for CORS (Cross-Origin Resource Sharing) using annotations like `@CrossOrigin`.

The code editor interface includes a toolbar at the top with icons for file operations, and a bottom navigation bar with tabs for other files like `login.html`, `sing.html`, `browse.html`, and `cart.html`.

```
35     @PostMapping("/signup")
36     @ResponseBody
37     public ResponseEntity<?> registerUser(@RequestBody User user) {
38         user.setPassword(passwordEncoder.encode(user.getPassword()));
39         userService.saveUser(user);
40         return ResponseEntity.ok( body: "User registered successfully");
41     }
42
43     no usages
44     @PostMapping("/login")
45     @ResponseBody
46     public ResponseEntity<?> authenticateUser(@RequestBody User loginUser) {
47         String username = loginUser.getUsername();
48         String password = loginUser.getPassword();
49
50         User user = userService.findByUsername(username);
51         if (user != null && passwordEncoder.matches(password, user.getPassword())) {
52             return ResponseEntity.ok(Map.of( k1: "message", v1: "User authenticated successfully"));
53         } else {
54             return ResponseEntity.status(401).body(Map.of( k1: "message", v1: "Invalid username or password"));
55         }
56     }
57
58     no usages
59     @PostMapping("/addproduct")
60     @ResponseBody
61     public ResponseEntity<?> addProduct(@RequestBody Product product) {
62         productService.saveProduct(product);
63         return ResponseEntity.ok( body: "Product added successfully");
64
65     no usages
66     @GetMapping("/getallproducts")
67     public ResponseEntity<List<Product>> getAllProducts(){
68         List<Product> products = productService.getAllProducts();
69         return new ResponseEntity<>(products, HttpStatus.OK);
70     }
71
72     no usages
73     @GetMapping("/getproductbyid")
74     public ResponseEntity<Product> getProductByID(@RequestParam String id){
75         Product product = productService.getProductById(id);
76         return new ResponseEntity<>(product, HttpStatus.OK);
77     }
78
79     no usages
80     @DeleteMapping("/deleteproductbyid")
81     public ResponseEntity<?> deleteProduct(@RequestParam String id){
82         productService.deleteProduct(id);
83         return ResponseEntity.ok( body: "Product deleted successfully");
84     }
85
86     no usages
```

```
① AuthController.java x ② CorsConfig.java    <> login.html    <> sing.html    <> browse.html    <> cart.html    ③ ProductService.java
```

```
82     no usages
83     @GetMapping("/search")
84     public ResponseEntity<List<Product>> searchProducts(@RequestParam String keyword) {
85         List<Product> products = productService.searchProducts(keyword);
86         return new ResponseEntity<>(products, HttpStatus.OK);
87     }
88     no usages
89     @GetMapping("/login")
90     public String showLoginPage() {
91         return "login";
92     }
93     no usages
94     @GetMapping("/signup")
95     public String showSignupPage() {
96         return "signup";
97     }
98     no usages
99     @PutMapping("/{id}")
100    public ResponseEntity<String> updateProduct(@PathVariable String id, @RequestBody Product productDetails) {
101        Optional<Product> updatedProduct = productService.updateProduct(id, productDetails);
102        if (updatedProduct.isPresent()) {
103            return new ResponseEntity<>("Product updated successfully", HttpStatus.OK);
104        } else {
105            return new ResponseEntity<>("Product not found", HttpStatus.NOT_FOUND);
106        }
107    }
108    no usages
109    @PostMapping("/cart") // To add the product into the cart
110    @ResponseBody
111    public ResponseEntity<String> addToCart(@RequestBody Map<String, String> request) {
112        String username = request.get("username");
113        String productId = request.get("productId");
114
115        // Implement logic to add product to cart for the user
116        // Example:
117        Optional<User> user = userService.addProductToCart(username, productId);
118        if (user.isPresent()) {
119            return ResponseEntity.ok("Product added to cart");
120        } else {
121            return ResponseEntity.status(HttpStatus.BAD_REQUEST).body("User not found");
122        }
123    }
124
125    no usages
126    @PostMapping("/kiosk") // To add the product into the KIOSK
127    @ResponseBody
128    public ResponseEntity<String> addToKiosk(@RequestBody Map<String, String> request) {
129        String username = request.get("username");
130        String productId = request.get("productId");
131
132        // Implement logic to add product to cart for the user
133        // Example:
134        Optional<User> user = userService.addProductToKiosk(username, productId);
135        if (user.isPresent()) {
136            return ResponseEntity.ok("Product added to Kiosk");
137        } else {
```

```

no usages
@DeleteMapping("/cart/remove/{productId}")
@ResponseBody
public ResponseEntity<String> removeFromCart(@PathVariable String productId, @RequestParam String username) {
    boolean isRemoved = userService.removeProductFromCart(username, productId);
    if (isRemoved) {
        return ResponseEntity.ok("Product removed from cart");
    } else {
        return ResponseEntity.status(HttpStatus.BAD_REQUEST).body("Product or user not found");
    }
}

no usages
@GetMapping("/cart/{username}")
public ResponseEntity<List<Product>> getCartItems(@PathVariable String username) {
    //String username = getUsernameFromSecurityContext();
    Optional<List<Product>> cartItems = userService.getCartItems(username);
    if (cartItems.isPresent()) {
        return ResponseEntity.ok(cartItems.get());
    } else {
        return ResponseEntity.status(400).body(null);
    }
}

//Cart count api for browse.html page
no usages
@GetMapping("/cart/count/{username}")
public ResponseEntity<Integer> getCartItemCount(@PathVariable String username) {
    int itemCount = userService.getCartItemCount(username);
    return ResponseEntity.ok(itemCount);
}

// Kiosk API's

no usages
@GetMapping("/kiosk/count")
public ResponseEntity<Integer> getKioskItemCount() {
    // Assuming you have a method in CartService that gets the count of items in the kiosk cart
    int itemCount = cartService.getCartItemCount();
    return ResponseEntity.ok(itemCount);
}

no usages
@GetMapping("/searchByName")
@ResponseBody
public ResponseEntity<List<Product>> searchProductsByName(@RequestParam String keyword) {
    List<Product> products = productService.searchProductsByName(keyword);
    return new ResponseEntity<>(products, HttpStatus.OK);
}

```

```
no usages
@PostMapping("/kioskcart")
@ResponseBody
public ResponseEntity<String> addKioskToCart(@RequestBody Map<String, String> request) {
    cartService.addToCart(request.get("productId"));
    return ResponseEntity.ok( body: "Product added to cart");
}

no usages
@DeleteMapping("/kioskcart/remove/{productId}")
@ResponseBody
public ResponseEntity<String> removeFromCart(@PathVariable String productId) {
    cartService.removeFromCart(productId);
    return ResponseEntity.ok( body: "Product removed from cart");
}

no usages
@GetMapping("/getkioskcart")
public ResponseEntity<List<Product>> getKioskCartItems() {
    List<Product> cartItems = cartService.getCartItems();
    return new ResponseEntity<>(cartItems, HttpStatus.OK);
}

no usages
@GetMapping("/kioskcart/count")
public ResponseEntity<Integer> getCartItemCount() {
    int itemCount = cartService.getCartItemCount();
    return ResponseEntity.ok(itemCount);
}

no usages
@GetMapping("/resetkioskcart")
public ResponseEntity<String> clearKioskCart(){
    cartService.clearCart();
    return ResponseEntity.ok( body: "Cart has been reset");
}

no usages
@PostMapping("/placeOrder/{username}")
public ResponseEntity<String> placeOrder(@PathVariable String username, @RequestBody Order order) {
    userService.placeOrder(username, order);
    //orderService.placeOrder(username, order);
    return ResponseEntity.ok( body: "Order Placed");
}
```

```

no usages
@GetMapping("orders/{username}")
public ResponseEntity<List<Order>> getOrders(@PathVariable String username) {
    List<Order> orders = orderService.getOrdersByUsername(username);
    return ResponseEntity.ok(orders);
}

no usages
@PostMapping("/placekioskorder")
public ResponseEntity<String> placeOrder(@RequestBody Order order) {
    orderService.saveOrder(order); // Assuming saveOrder method is implemented in the OrderService
    return ResponseEntity.ok( body: "Order Placed Successfully");
}

// browser clear cart: confirmation page ---> browser page

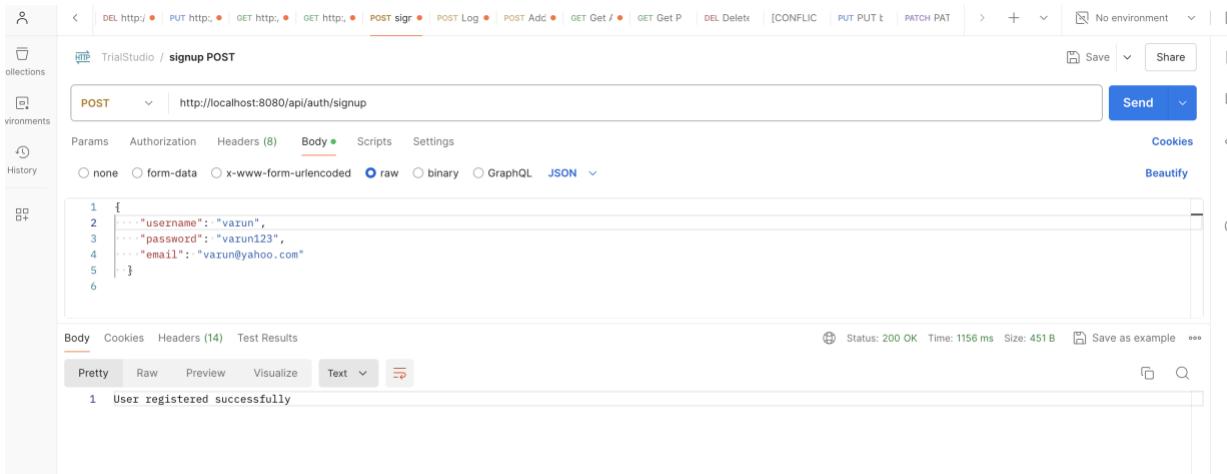
no usages
@PostMapping("/clearcart")
public ResponseEntity<Void> clearCart(@RequestBody Map<String, String> request) {
    String username = request.get("username");
    User user = userService.findByUsername(username);
    if (user != null) {
        user.getCart().clear();
        userService.save(user); // Save the user to update the cart in the database
        return ResponseEntity.ok().build();
    } else {
        return ResponseEntity.status(HttpStatus.NOT_FOUND).build();
    }
}

// kiosk clear cart: kioskconfirmation page ---> kioskbrowser page
no usages
@PostMapping("/resetkioskcart")
public ResponseEntity<Void> resetKioskCart() {
    cartService.clearCart(); // Assuming clearCart is a method in cartService that clears the kiosk cart
    return ResponseEntity.ok().build();
}

}

```

## 12. API response on Postman:



HTTP TrialStudio / signup POST

POST http://localhost:8080/api/auth/signup

Params Authorization Headers (8) Body Scripts Settings

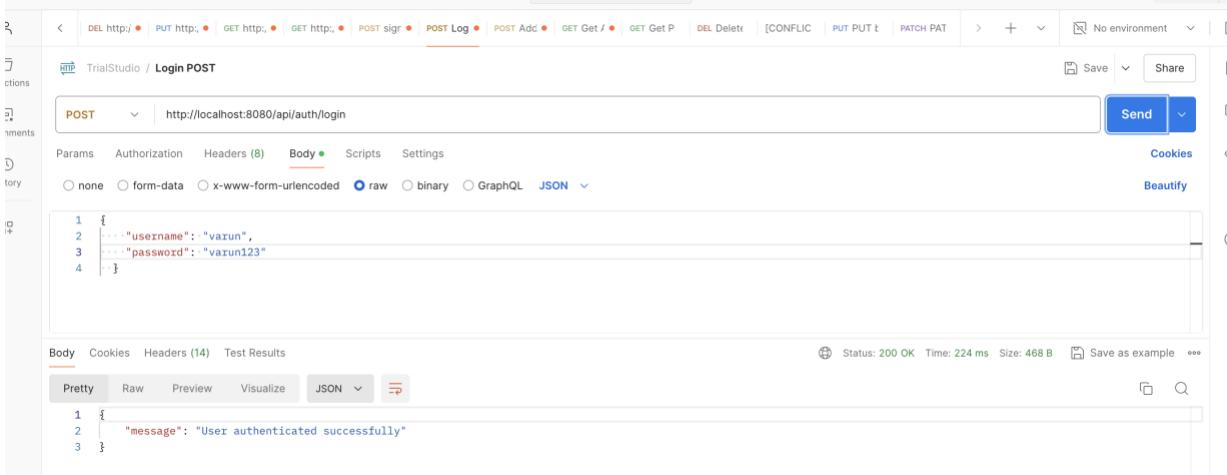
Body (Pretty, Raw, Preview, Visualize, Text)

```
1 {
2   "username": "varun",
3   "password": "varun123",
4   "email": "varun@yahoo.com"
5 }
```

Body Cookies Headers (14) Test Results

Status: 200 OK Time: 1156 ms Size: 451 B Save as example

1 User registered successfully



HTTP TrialStudio / Login POST

POST http://localhost:8080/api/auth/login

Params Authorization Headers (8) Body Scripts Settings

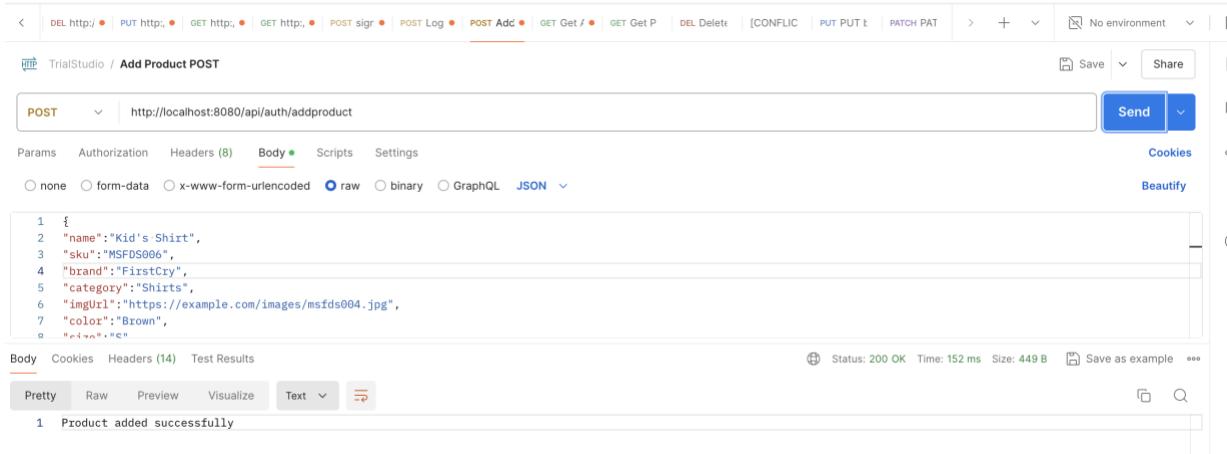
Body (Pretty, Raw, Preview, Visualize, JSON)

```
1 {
2   "username": "varun",
3   "password": "varun123"
4 }
```

Body Cookies Headers (14) Test Results

Status: 200 OK Time: 224 ms Size: 468 B Save as example

1 {
2 "message": "User authenticated successfully"
3 }



HTTP TrialStudio / Add Product POST

POST http://localhost:8080/api/auth/addproduct

Params Authorization Headers (8) Body Scripts Settings

Body (Pretty, Raw, Preview, Visualize, JSON)

```
1 {
2   "name": "Kid's Shirt",
3   "sku": "MSFDS006",
4   "brand": "FirstCry",
5   "category": "Shirts",
6   "imgUrl": "https://example.com/images/msfds004.jpg",
7   "color": "Brown",
8 }
```

Body Cookies Headers (14) Test Results

Status: 200 OK Time: 152 ms Size: 449 B Save as example

1 Product added successfully

[DEL http://](#) | [PUT http://](#) | [GET http://](#) | [POST http://](#) | [POST sign](#) | [POST Log](#) | [POST Add](#) | [GET Get F](#) | [GET Get P](#) | [DEL Delete](#) | [\[CONFLICT\]](#) | [PUT PUT t](#) | [PATCH PAT](#) | > | + | [No environment](#) | [Save](#) | [Share](#)

### TrialStudio / Get All Products GET

GET http://localhost:8080/api/auth/getallproducts

Params Authorization Headers (6) Body Scripts Settings Cookies

Status: 200 OK Time: 118 ms Size: 1.5 KB Save as example

Body Cookies Headers (14) Test Results

Pretty Raw Preview Visualize JSON

```

1 [
2   {
3     "id": "668c92f32d5e5f5a850b6e99",
4     "name": "Women's Top",
5     "sku": "NEW-SKU-67899",
6     "brand": "Levis",
7     "category": "Trousers",
8     "imgUrl": null,
9     "color": "Orange",
10    "size": "XL",
11    "description": "Updated product description",
12    "availability": true
13  },
14  {
15    "id": "668dd2a605e6437d05177ecd",
16    "name": "Men's Shirt",
17    "sku": "MSFDS003",
18    "brand": "Reebok",
19    "category": "Shirts",
20    "imgUrl": "https://example.com/images/msfds003.jpg",
21    "color": "Grey",
22    "size": "XL",
23    "description": "Classic slim-fit dress shirt for men, suitable for formal occasions.",
24    "availability": true
25  }
]

```

Find and replace Console Postbot Runner Start Proxy Cookies Vault Trash

[DEL http://](#) | [PUT http://](#) | [GET http://](#) | [POST http://](#) | [POST sign](#) | [POST Log](#) | [POST Add](#) | [GET Get F](#) | [GET Get P](#) | [DEL Delete](#) | [\[CONFLICT\]](#) | [PUT PUT t](#) | [PATCH PAT](#) | > | + | [No environment](#) | [Save](#) | [Share](#)

### TrialStudio / Get Product by ID GET

GET http://localhost:8080/api/auth/getproductbyid?id=66921eb9d96b633672c5b250

Params Authorization Headers (6) Body Scripts Settings Cookies

Status: 200 OK Time: 107 ms Size: 716 B Save as example

Query Params

Key	Value	Description	... Bulk Edit
<input checked="" type="checkbox"/> id	66921eb9d96b633672c5b250		
Key	Value	Description	

Body Cookies Headers (14) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": "66921eb9d96b633672c5b250",
3   "name": "Kid's Shirt",
4   "sku": "MSFDS006",
5   "brand": "FirstCry",
6   "category": "Shirts",
7   "imgUrl": "https://example.com/images/msfds004.jpg",
8   "color": "Brown",
9   "size": "S",
10  "description": "Classic slim-fit dress shirt for men, suitable for formal occasions.",
11  "availability": true
12 }

```

**DELETE** http://localhost:8080/api/auth/deleteproductbyid?id=668c92f32d5e5f5a850b6e99

**Params** Authorization Headers (6) Body Scripts Settings Cookies

**Query Params**

Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> id	668c92f32d5e5f5a850b6e99		
Key	Value	Description	

**Body** Cookies Headers (14) Test Results

Status: 200 OK Time: 121 ms Size: 451 B Save as example

Pretty Raw Preview Visualize Text

```
1 Product deleted successfully
```

---

**GET** http://localhost:8080/api/auth/search?keyword=FirstCry

**Params** Authorization Headers (6) Body Scripts Settings Cookies

**Query Params**

Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> keyword	FirstCry		
Key	Value	Description	

**Body** Cookies Headers (14) Test Results

Status: 200 OK Time: 93 ms Size: 718 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": "66921eb96b633672c5b250",
4     "name": "Kid's Shirt",
5     "sku": "MSFD5006",
6     "brand": "FirstCry",
7     "category": "Shirts",
8     "imgUrl": "https://example.com/images/msfds004.jpg",
9     "color": "Brown",
10    "size": "S",
11    "description": "Classic slim-fit dress shirt for men, suitable for formal occasions.",
12    "availability": true
13  }
14 ]
```

HTTP TrialStudio / PUT by ID

PUT http://localhost:8080/api/auth/668dd2a605e6437d05177ecd

Params Authorization Headers (8) Body Scripts Settings

Body (Pretty) Content-Type: application/json

```
1 {  
2   "name": "Men's slim-fit Shirt",  
3   "sku": "MSFDS003",  
4   "brand": "Nike",  
5   "category": "Shirts",  
6   "imgurl": "https://example.com/images/msfds003.jpg",  
7   "color": "Black",  
8   "size": "L",  
9   "description": "Classic slim-fit dress shirt for men, suitable for formal occasions.",  
10  "availability": true  
11}
```

Body Cookies Headers (14) Test Results

Pretty Raw Preview Visualize Text Status: 200 OK Time: 211 ms Size: 451 B Save as example

```
1 Product updated successfully
```

HTTP TrialStudio / PATCH by ID

PATCH http://localhost:8080/api/auth/668dd2a605e6437d05177ecd

Params Authorization Headers (8) Body Scripts Settings

Body (Pretty) Content-Type: application/json

```
1 {  
2   "name": "Men's slim-fit T-Shirt",  
3   "sku": "NEW-SKU-67899"  
4 }  
5
```

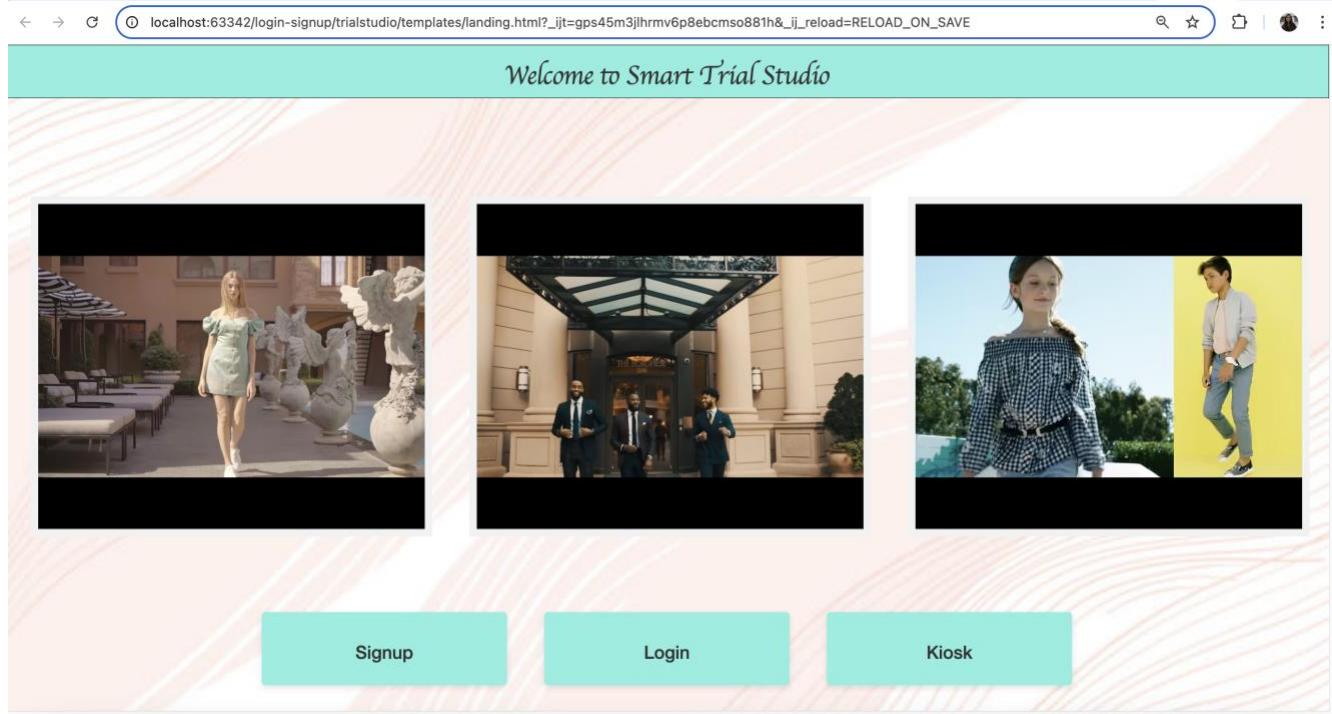
Body Cookies Headers (14) Test Results

Pretty Raw Preview Visualize Text Status: 200 OK Time: 199 ms Size: 451 B Save as example

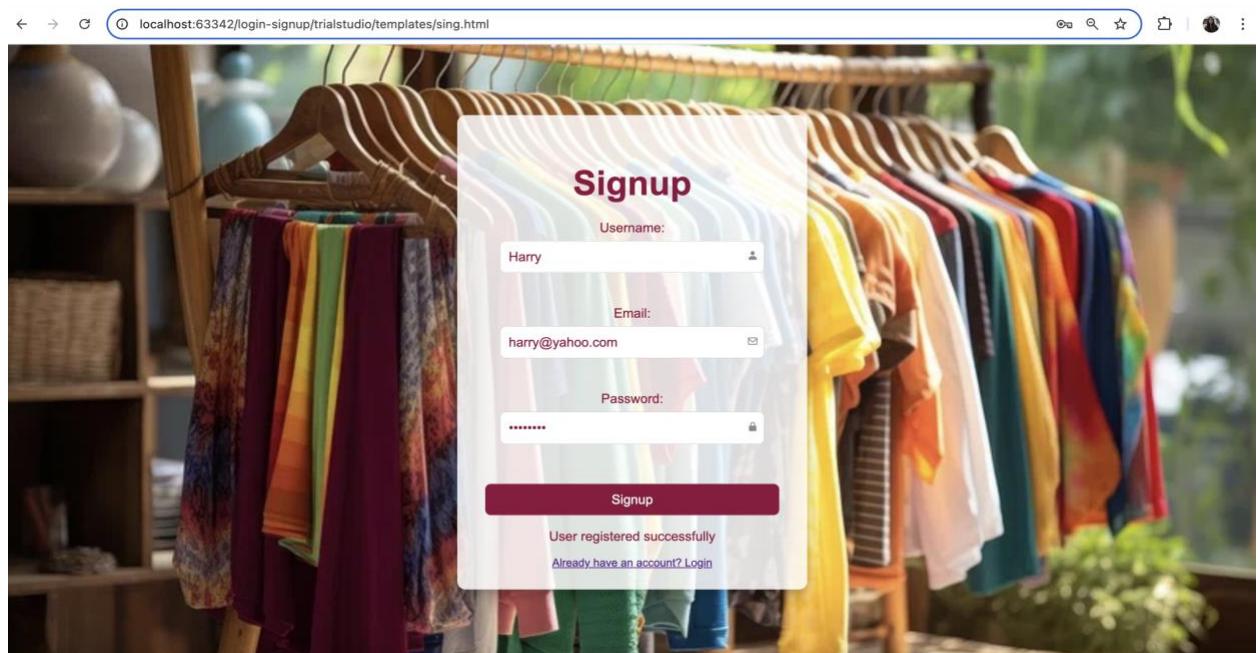
```
1 Product updated successfully
```

## 13. Project Execution:

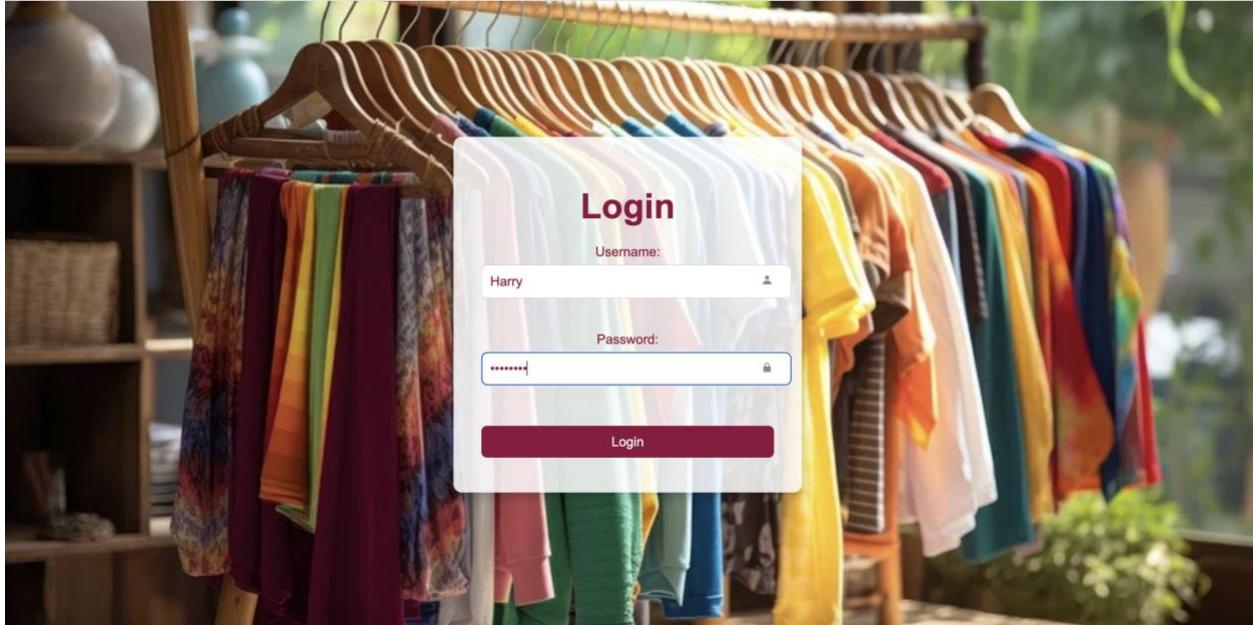
### 1. Landing Page:



### 2. Sing Up page:



### 3. Login Page:



### 4. Browser page:

A screenshot of a web browser displaying the "SMART TRIAL STUDIO" website. The top navigation bar includes links for "Logout", "View Orders", and "Cart: 0 items". The main content area features a grid of six product cards. Top row: 1. Men's slim-fit T-Shirt (black, L, \$22) with an "Add to Cart" button. 2. Women's slim-fit Shirt (brown, S, \$14) with an "Add to Cart" button. 3. Kid's Shirt (brown, S, \$25) with an "Add to Cart" button. Bottom row: 1. Kid's Trouser (white, XS, \$32) with an "Add to Cart" button. 2. Leather Shoulder Bag (pink, M, \$69) with an "Add to Cart" button. 3. Women's Cotton T-Shirt (red, L, \$18) with an "Add to Cart" button. Each card contains a small product image, a brief description, brand, category, color, size, and price.

localhost:63342/login-signup/trialstudio/templates/browse.html

**Women Beauty Nourishing Lip Balm**

Color: Silver  
Size: One Size  
Price: 85

**Male Suits Classic Navy Blue Suit**

A classic navy blue suit featuring a tailored fit and a timeless design. Perfect for formal events and professional settings.

Brand: Elegance Tailors  
Category: Suits  
Color: Navy Blue  
Size: M  
Price: 349

**Male Suits Gray Checkered Suit**

A stylish gray checkered suit with a slim fit, ideal for both business and casual occasions. Features a modern design with a checkered pattern.

Brand: Modern Fit  
Category: Suits  
Color: Gray  
Size: L  
Price: 399

**Male Suits Black Tuxedo Suit**

An elegant black tuxedo suit, perfect for formal events and evening wear. Features a classic design with satin lapels and a tailored fit.

Brand: Luxury Line  
Category: Suits  
Color: Black  
Size: XL  
Price: 400

**Male Suits Charcoal Grey Suit**

A sophisticated charcoal grey suit with a contemporary cut, ideal for professional settings and formal occasions.

Brand: Executive Wear  
Category: Suits  
Color: Charcoal Grey  
Size: S  
Price: 359

**Male Suits Light Beige Linen Suit**

A light beige linen suit designed for warmer weather. Features a breathable fabric and relaxed fit, perfect for summer events.

Brand: Summer Style  
Category: Suits  
Color: Beige  
Size: M  
Price: 275

## 5. Cart page

SMART TRIAL STUDIO

Logged in as Harry

No of Items: 2

**Your Cart**

	<b>Male Suits Classic Navy Blue Suit</b> A classic navy blue suit featuring a tailored fit and a timeless design. Perfect for formal events and professional settings. Price: \$349	<input type="button" value="Remove"/>
	<b>Male Shirt Short Sleeve Plaid</b> A short sleeve plaid shirt. Made from soft, breathable fabric with a relaxed fit. Perfect for casual summer outings. Price: \$30	<input type="button" value="Remove"/>

Total Price: \$379.00

## **6. Checkout page:**

**Checkout**  
Logged in as Harry

**Billing Details**

First Name \*

Last Name \*

Company Name (optional)

Country/Region \*

Street Address \*

Town/City \*

State/County (optional)

Postcode/ZIP (optional)

Phone \*

**Payment Options**

Name on Card

Card Number

Expiration

CVV

**Phone \***

**Email Address \***

**Your Order**

<b>Product</b>	<b>Subtotal</b>
Male Suits Classic Navy Blue Suit x 1	\$349.00
Male Shirt Short Sleeve Plaid x 1	\$30.00
Shipping	\$10.00
<b>Total</b>	<b>\$389.00</b>

## 7. Order confirmation

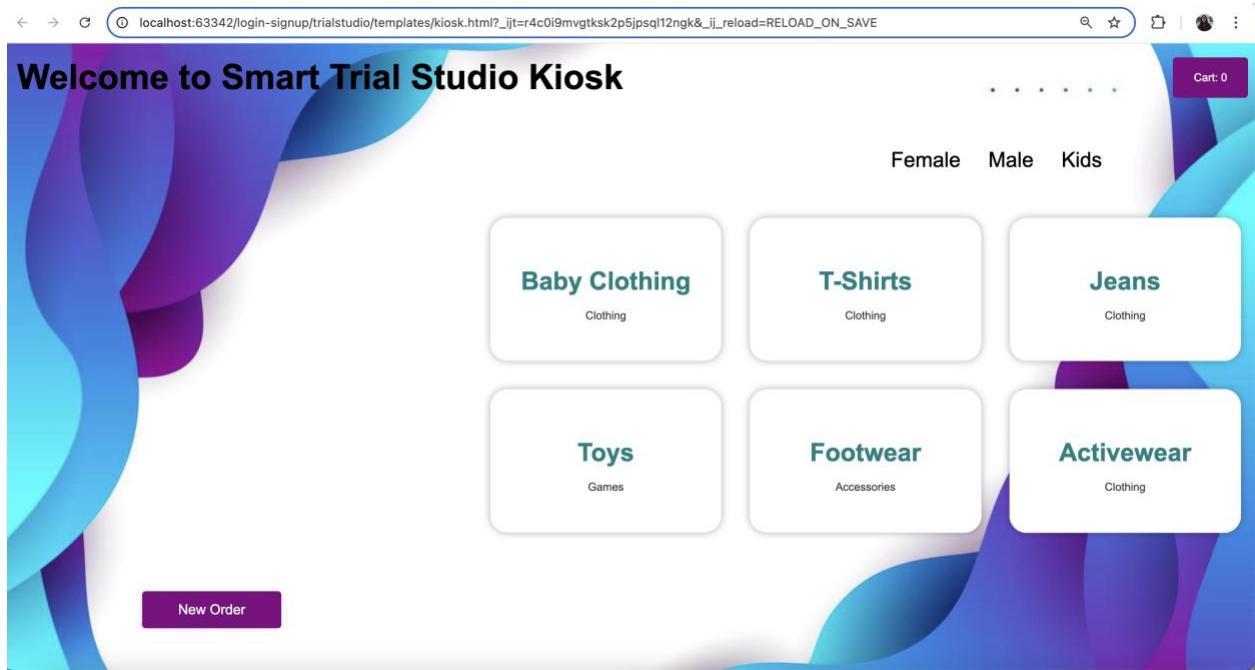
The image consists of two vertically stacked screenshots of a web browser window.

**Screenshot 1 (Top):** This screenshot shows a checkout form on a page titled "localhost:63342/login-signup/trialstudio/templates/checkout.html". The form includes fields for "Country/Region" (United States), "Street Address" (235 Boardwalk hamilton street), "Town/City" (San Jose), and "State/County (optional)" (CA). A modal dialog box is overlaid on the page, displaying the message "localhost:63342 says" followed by "Order placed successfully!" and a blue "OK" button.

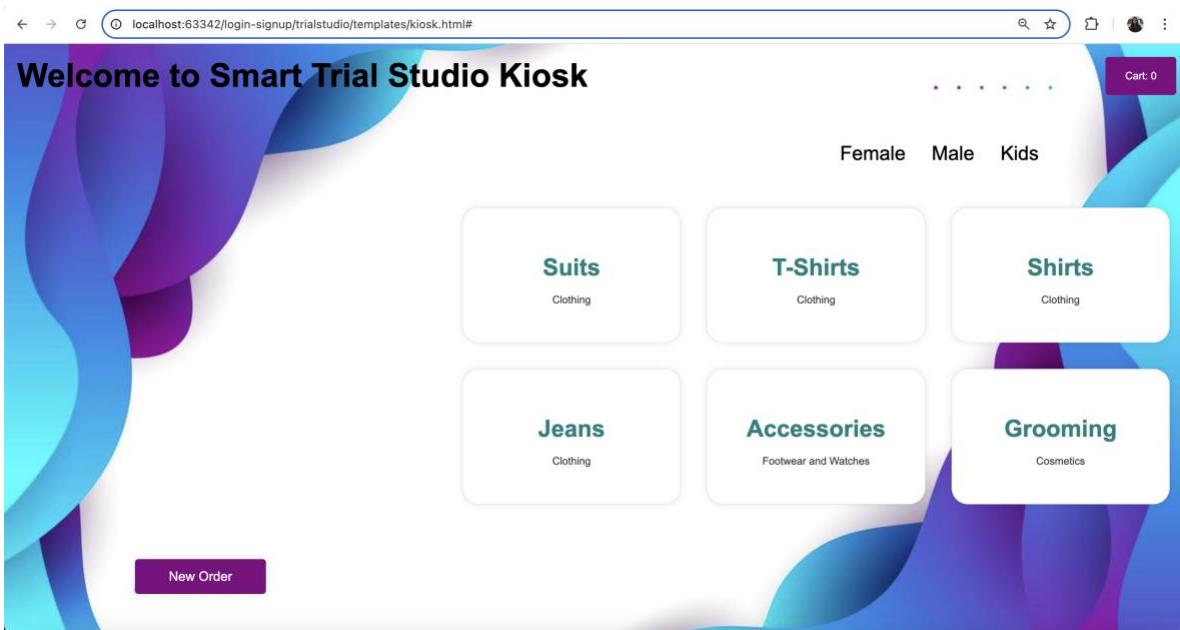
**Screenshot 2 (Bottom):** This screenshot shows a confirmation page on a page titled "localhost:63342/login-signup/trialstudio/templates/confirmation.html". The main heading is "Thank You for Your Order!". Below it, a message states "Your order has been placed successfully." and "We appreciate your business." There is a green "Continue Shopping" button. At the bottom of the page, a footer bar contains the text "© 2024 Smart Trial Studio".

## 8. Kiosk Page

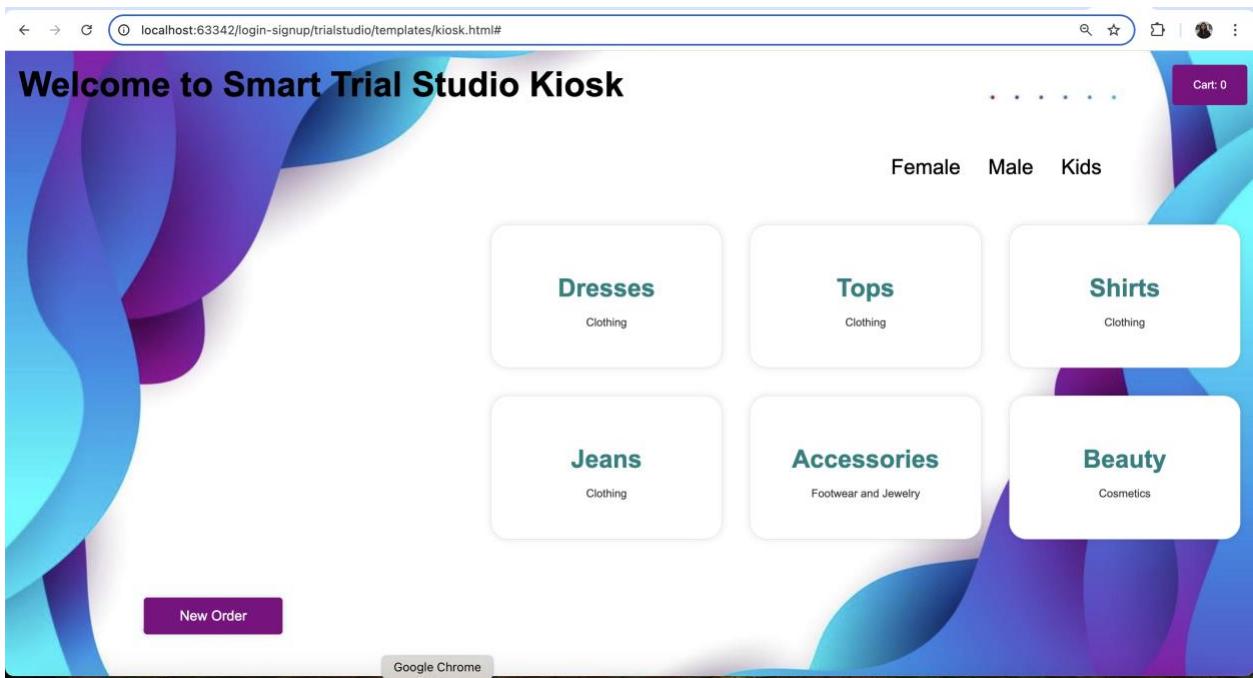
### a) Kids:



### b) Male:



c) **Female:**



**9. Browse products on kiosk:**

The screenshot shows a grid of six dress products under the heading 'SMART TRIAL STUDIO'. Each product card includes a thumbnail image, the name, a brief description, brand, category, color, size, price, and an 'Add to Cart' button.

Product	Description	Brand	Category	Color	Size	Price
Boho Maxi Dress	A free-flowing maxi dress with intricate embroidery and a relaxed fit. Ideal for a bohemian look.	Bohemian Style	Maxi Dresses	White	L	45
Cocktail Party Dress	A knee-length dress with a bodycon fit and a deep V-neckline. Perfect for cocktail parties and night outs.	Night Out	Cocktail Dresses	Navy Blue	S	25
Vintage Polka Dot Dress	A classic vintage dress with a polka dot pattern, cinched waist, and A-line skirt.	Retro Vibes	Vintage	Black and White	M	20
Elegant Lace Dress	A delicate lace dress with a fitted bodice and a flared skirt. Ideal for formal events and weddings.	Lace & Grace	Formal	Ivory	XL	50
Sporty Tank Dress	A comfortable tank dress with a sporty design. Perfect for casual wear and outdoor activities.	Active Wear	Casual	Gray	M	38
Romantic Ruffle Dress	A romantic dress featuring ruffles and a soft, flowy fabric. Perfect for date nights and special occasions.	Romantic Flair	Party	Pink	S	29

localhost:63342/login-signup/trialstudio/templates/kioskBrowse.html?category=top

# SMART TRIAL STUDIO

Search

Cart: 2 items

### Boho Lace Top

A bohemian lace top with intricate detailing and a flowy silhouette, great for a relaxed, stylish look.

**Brand:** Bohemian Chic  
**Category:** Blouses  
**Color:** White  
**Size:** L  
**Price:** 45



[Add to Cart](#)

### Classic woment top

A versatile teal top with a button-down front, made from lightweight, breathable fabric. Perfect for both professional and casual settings.

**Brand:** Elegant Essentials  
**Category:** Tops  
**Color:** Teal  
**Size:** M  
**Price:** 35



[Add to Cart](#)

### Casual Stripe Top

A casual top featuring bold stripes and a comfortable fit, perfect for everyday wear.

**Brand:** Urban Trend  
**Category:** T-Shirts  
**Color:** Sky Blue  
**Size:** S  
**Price:** 30



[Add to Cart](#)

### Luxe Satin Top

A luxurious satin top with a sleek design and adjustable straps, ideal for elegant evening wear.

**Brand:** Elegance Wear  
**Category:** Blouses  
**Color:** Champagne  
**Size:** M  
**Price:** 45



[Add to Cart](#)

### High Neck Top

A chic high-neck top made from soft cotton, ideal for layering or wearing alone for a sleek look.

**Brand:** Modern Classic  
**Category:** T-Shirts  
**Color:** Green  
**Size:** XS  
**Price:** 35



[Add to Cart](#)

### Graphic Print Top

A trendy graphic print top with a bold design, perfect for adding a pop of style to casual outfits.

**Brand:** Trendsetter  
**Category:** Graphic Tees  
**Color:** Red  
**Size:** M  
**Price:** 28



[Add to Cart](#)

SMART TRIAL STUDIO

localhost:63342 says  
Product added to cart

**Cream**

A rich and creamy shaving cream that provides a smooth shave. Enriched with moisturizers to protect and soothe the skin.

Brand: Smooth Shave

Category: Grooming

Color: White

Size: 100g

Price: 12

OK

**Male Grooming Beard Oil**

A nourishing beard oil designed to hydrate and condition your beard. Infused with natural oils and vitamins to promote healthy beard growth.

Brand: Beard Essentials

Category: Grooming

Color: Clear

Size: 30ml

Price: 18

**Male Grooming Hair Styling Gel**

A high-hold hair styling gel that keeps your hairstyle in place all day. Non-greasy and easy to wash out.

Brand: Urban Style

Category: Grooming

Color: Clear

Size: 150ml

Price: 15

Add to Cart

Add to Cart

Cart: 2 items

## 10. Kiosk cart

The screenshot shows a web browser window with the URL `localhost:63342/login-signup/trialstudio/templates/kioskCart.html`. The page has a header "SMART TRIAL STUDIO" and a top right corner showing "No of Items: 4". The main content area is titled "Your Cart" and lists four items:

- Romantic Ruffle Dress**: A romantic dress featuring ruffles and a soft, flowy fabric. Perfect for date nights and special occasions.  
Price: \$29 Remove
- Casual Stripe Top**: A casual top featuring bold stripes and a comfortable fit, perfect for everyday wear.  
Price: \$30 Remove
- Male Grooming Shaving Cream**: A rich and creamy shaving cream that provides a smooth shave. Enriched with moisturizers to protect and soothe the skin.  
Price: \$12 Remove
- Baby Toys Activity Gym Mat**: An engaging activity gym mat with hanging toys to stimulate your baby's senses. Soft and padded for comfort.  
Price: \$35 Remove

Total Price: \$106.00 [Proceed to Checkout](#)

## 11. Kiosk Checkout

The screenshot shows a web browser window with the URL `localhost:63342/login-signup/trialstudio/templates/kioskCheckout.html`. The page has a header "Kiosk Checkout". It is divided into two main sections: "Billing Details" on the left and "Payment Options" on the right.

**Billing Details**

First Name \*

Last Name \*

Company Name (optional)

Country/Region \*

Street Address \*

Town/City \*

State/County (optional)

Postcode/ZIP (optional)

Phone \*

**Payment Options**

Name on Card

Card Number

Expiration

CVV

Email Address \*

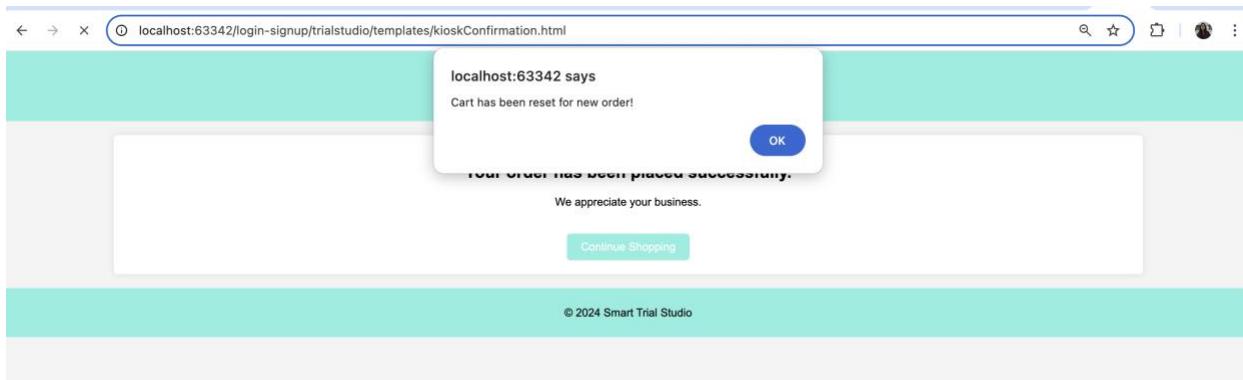
**Your Order**

Product	Subtotal
Romantic Ruffle Dress x 1	\$29.00
Casual Stripe Top x 1	\$30.00
Male Grooming Shaving Cream x 1	\$12.00
Baby Toys Activity Gym Mat x 1	\$35.00
<b>Total</b>	<b>\$106.00</b>

[Place Order](#)

© 2024 Smart Trial Studio

## 12. Order Confirmation:



## 14. Challenges:

- A. **Mongodb connection:** I faced some issues when trying to connect to MongoDB, which is the database I used for the project. The connection kept failing due to problems like incorrect settings and outdated software versions. After some troubleshooting, I figured out the issue was with the network configuration and the database driver. Once I fixed these, the connection worked, and I could continue with the project.

**B. Security Configs:** I had trouble with the security settings when connecting to MongoDB. The database kept blocking my access because the security configurations were too strict. After adjusting the permissions and making sure my credentials were correct, I was finally able to connect and move forward with the project.

**C. API execution:** I ran into issues while creating the REST API for my project. The problem was that some of the endpoints weren't working correctly because of mistakes in how I set up the routes and handled requests. After debugging and fixing the code, the API started functioning as expected, and I could connect it to the rest of the application.

## 15. Completed Tasks:

### Steps I followed to create the project:

1. Setting up MongoDB from <https://www.mongodb.com/cloud/atlas/register>
2. Created Spring boot project structure from <https://start.spring.io/>
3. Created API in AuthController.java
4. Used POSTMAN to test all the API's
5. Created web pages for Signup, Login, Browser and Cart, Checkout, Confirmation, Kiosk home page, kiosk browser page, Kiosk cart, Kiosk checkout, kiosk confirmation.
6. Added many functionalities to make the API and web pages execute smoothly.

**PLEDGE:**

**I have neither given nor received unauthorized aid in completing this work, nor have I presented someone else's work as my own.**