

# TECHNOLOGY



## Project Session 02: Admin Dashboard Web Pages and Database Structure

# You Already Know

Before we begin, let's recall what we have covered so far:

- Git



- CSS



- SQL



- Angular



- HTML



# A Day in the Life of a Full Stack Developer

After syncing Angular projects on GitHub, Bob wants to design the frontend, and create a database structure for his web application.

Let me think about this. Which technologies should I use for these requirements?



# A Day in the Life of a Full Stack Developer

After brainstorming with the team and doing his own research, Bob found a solution for his requirements.

I will use HTML and CSS to design and develop web pages and then SQL and MySQL to design the database structure and create tables.



In this lesson, we will learn how to design, develop, and create a database structure for web pages to help Bob complete his task effectively and quickly.



## Learning Objectives

By the end of this lesson, you will be able to:

- 👁️ Develop the web page templates for admin dashboard
- 👁️ Develop the CSS for styling the web pages
- 👁️ Apply Angular component templates to build the structure of the frontend
- 👁️ Correlate the pages in Angular with routing for seamless navigation



## Learning Objectives

- 👁 Design a database for your project in MySQL
- 👁 Design a web app project for end user using Angular CLI
- 👁 Create primary and foreign key relations within tables for a robust database structure



## Develop the Web Pages in Angular for the Admin Dashboard

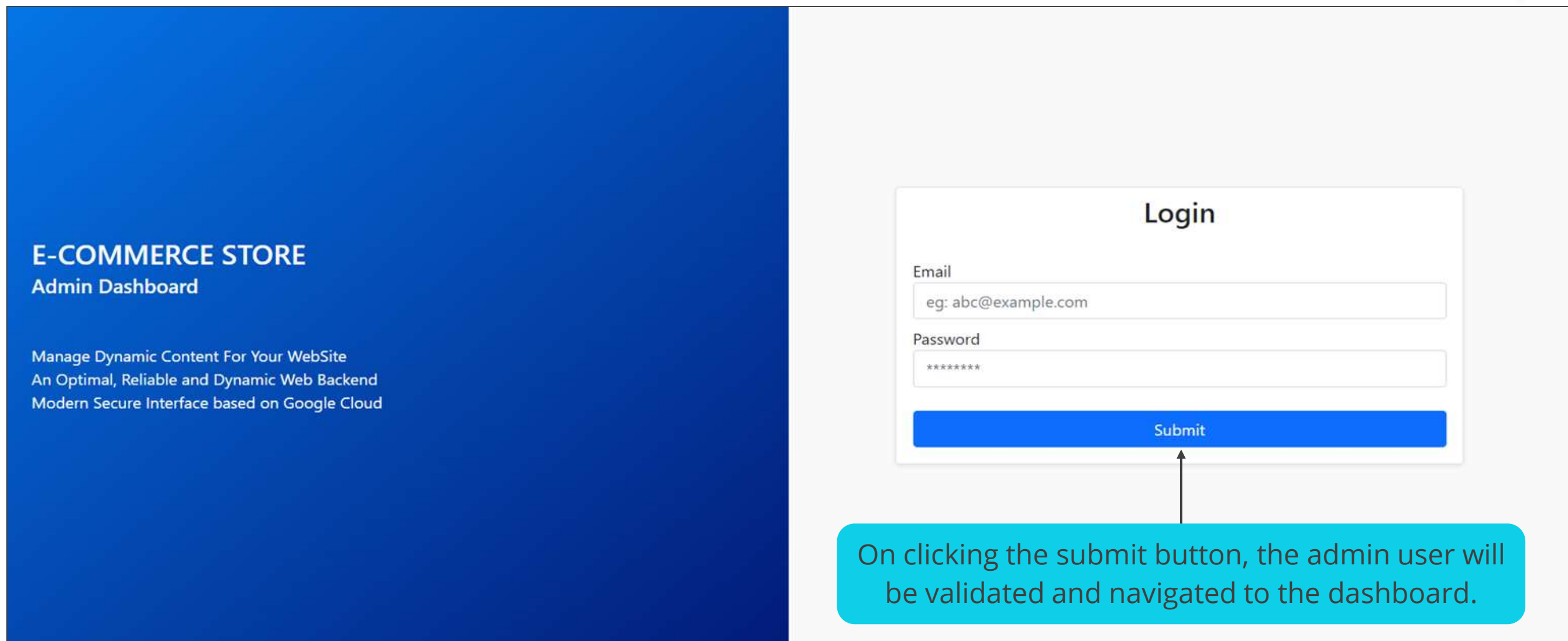


## Angular Components



# Web Page for Authentication Login Component

To begin, let's develop a login page for admin to sign in and access the dashboard for various management flows.



**E-COMMERCE STORE**  
Admin Dashboard

Manage Dynamic Content For Your WebSite  
An Optimal, Reliable and Dynamic Web Backend  
Modern Secure Interface based on Google Cloud

### Login

Email  
eg: abc@example.com

Password  
\*\*\*\*\*

Submit

On clicking the submit button, the admin user will be validated and navigated to the dashboard.

# Web Page for Authentication Login Component

In the directory: **src/app/pages/login/**

Command	Use
login.component.css	CSS goes here to design the page and forms.
login.component.html	HTML code is written in this template file.
login.component.ts	Logic will be written in TypeScript file.

# Web Page for Users Component

On the admin dashboard, a navigation bar should be designed to allow users to navigate between various modules. The users' component should have a list of all the users registered with the platform.

E-COMMERCE STORE Users Products Orders Shipments Payments Logout						
Users						
Sr. No	Name	Username	Email	City	Contact	Actions
1	John Watson	john	john@example.com	Ludhiana	9874563211	View Orders

The search can be implemented through phone number or email ID.

The view order button shows the order history of the corresponding user.

# Web Page for Users Component

In the directory: src/app/pages/users/

Command	Use
users.component.css	CSS goes here to design the page and forms.
users.component.html	HTML code is written in this template file.
users.component.ts	Logic will be written in TypeScript file.



# Web Page for Products Component

To display the products along with categories, the admin must add them from the dashboard.

The screenshot shows the 'Products' page of an 'E-COMMERCE STORE'. The top navigation bar includes links for Users, Products, Orders, Shipments, Payments, and Logout. Below the navigation bar, there are two buttons: 'Manage Product Categories' and 'Add Product'. The main content area displays a table of products. The table has columns for Sr. No, Product Code, Product Name, Description, Price, and Actions. A single product is listed: SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s, priced at ₹300. The Actions column for this product contains buttons for View Images, View Details, Update, and Delete. Annotations with arrows point to these elements: 'Manage Product Categories' is described as directing the admin to a category management component; 'Add Product' is described as opening a form to add products to the database; the product table is described as the place where all products are listed; and the action buttons are described as performing actions corresponding to their labels.

This button directs the admin to a category management component.

This button opens a form to add products to the database.

E-COMMERCE STORE Users Products Orders Shipments Payments Logout

Products

Sr. No	Product Code	Product Name	Description	Price	Actions
1	PRD-ELE-1054	SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s	Easy upgrade for faster boot up, shutdown, application load and response	₹300	View Images View Details Update Delete

All the products are listed here.

These buttons perform actions corresponding to their labels.

# Web Page for Products Component

In the directory: src/app/pages/products/

Command	Use
products.component.css	CSS goes here to design the page and forms.
products.component.html	HTML code is written in this template file.
products.component.ts	Logic will be written in TypeScript file.

# Category Management Component

In this web page, the UI should be developed to manage various categories for the products.

The Add Category button will open a modal to add the details to the database.

Sr. No	Name	Description	Status	Actions
1	Electronics	The field of electronics is a branch of physics and electrical engineering that deals with the emission, behaviour and effects of electrons using electronic devices.	Active	<a href="#">View Image</a> <a href="#">Update</a> <a href="#">Delete</a>

This page lists all the categories along with the action buttons to view, update, and delete the categories.

# Category Management Component

In the directory: `src/app/pages/categories/`

Command	Use
<code>categories.component.css</code>	CSS goes here to design the page and forms.
<code>categories.component.html</code>	HTML code is written in this template file.
<code>categories.component.ts</code>	Logic will be written in TypeScript file.



# Category Management Component: Add Category Modal

- Use the NgbModal service to develop the modal and add the category details.
- Create the modal for product category details in the directory `src/app/pages/modals/category.ts`

The screenshot displays an 'E-COMMERCE STORE' interface with a navigation bar (Users, Products, Orders, Shipments, Payments, Logout) and a 'Categories' table. A modal titled 'Product Category Details' is open, allowing for the addition of a new category. The modal includes a title field, a description text area, a file selection button, and a status toggle. A red bracket highlights the modal form, and a callout box explains its purpose.

Sr. No	Name	Description
1	Electronics	The field of electronics is a branch of physics and electrical engineering that deals with the emission, behaviour and effects of electrons using electronic devices.

**Product Category Details**

Title: Electronics

Description: The field of electronics is a branch of physics and electrical engineering that deals with the emission, behaviour and effects of electrons using electronic devices.

Select Image File: Choose File No file chosen

Status: Active In-Active

Cancel Save

Add Category

In this form, all the details of the product category are included.

# Web Page for Product Component: Add Product Modal

The action button should be used to add the product to the database on the products component web page.



On clicking, it should open a modal, which is a UI in a dialog view. Here, the user should be able to add the details of a product in a form.



The user should also be able to select the category from a dropdown to be linked to the product.



Here, the user should be able to associate as many images as they wish to link to the product, and these images must be shown as a thumbnail that can be managed.

# Web Page for Product Component: Add Product Modal

Below is an image of the product update page:

E-COMMERCE STORE

Products

Sr. No	Product Code
1	PRD-ELE-1054

Products

Sr. No	Product Code
1	PRD-ELE-1054

Product Update

Product Name

SanDisk SSD PLUS 1TB Internal SSD - SATA III €

Product Category

Electronics

Price

300

Product status

Enable

Disable

Description

Easy upgrade for faster boot up, shutdown, application load and response

Product Images

1. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX466\_.jpg

Selected as Thumbnail Image

Selected

Remove

2. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX679\_.jpg

Select

Remove

3. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX355\_.jpg

Select

Remove

4. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX522\_.jpg

Select

Remove

5. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX450\_.jpg

Select

Remove

6. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX425\_.jpg

Select

Remove

7. https://m.media-amazon.com/images/I/71J4Q8zM72L\_SX569\_.jpg

Select

Remove

Add

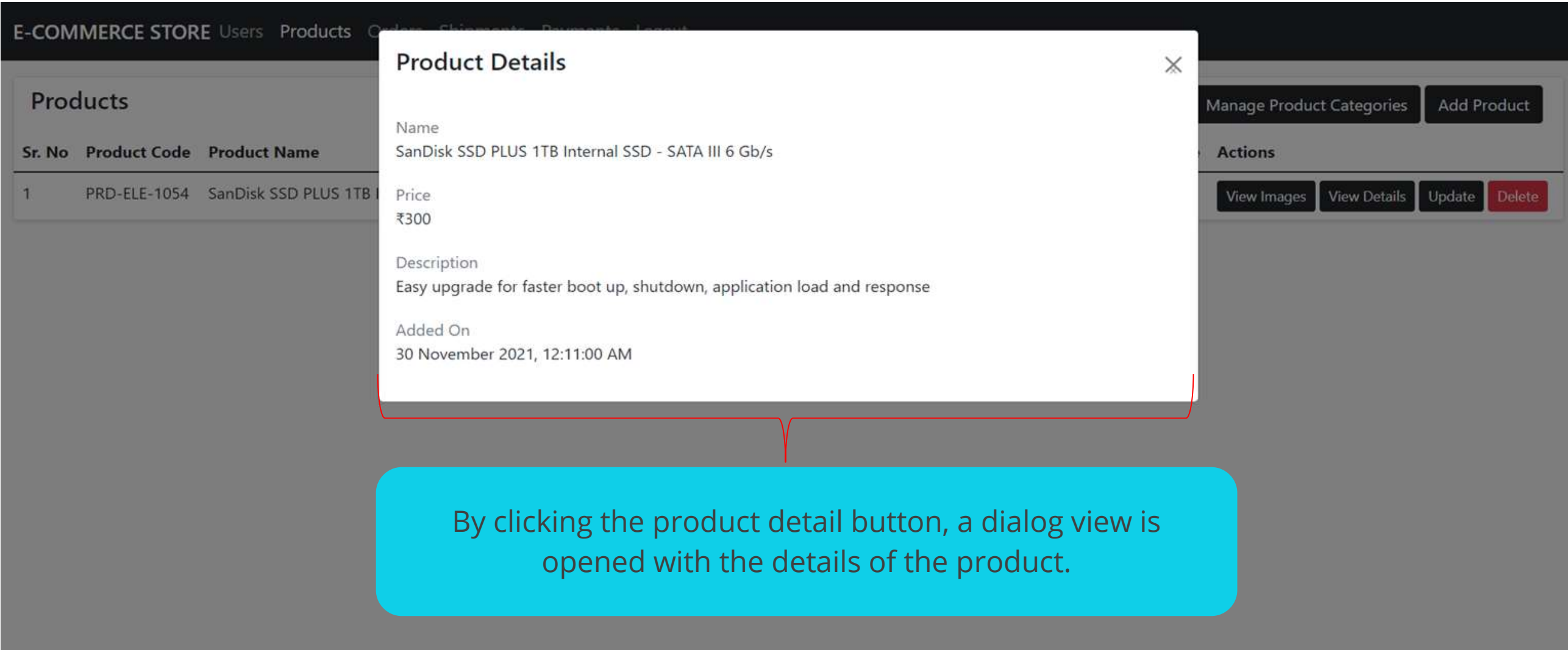
Cancel

Save

In this form, all the images related to the product can be added.

# Web Page for Product Component: Product Details Modal

Create the modal for adding product details in the directory:  
`src/app/pages/modals/products.ts`



The screenshot displays an 'E-COMMERCE STORE' interface. A 'Product Details' modal is open, showing the following information:

- Name:** SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s
- Price:** ₹300
- Description:** Easy upgrade for faster boot up, shutdown, application load and response
- Added On:** 30 November 2021, 12:11:00 AM

The background interface includes a 'Products' table with the following data:

Sr. No	Product Code	Product Name
1	PRD-ELE-1054	SanDisk SSD PLUS 1TB

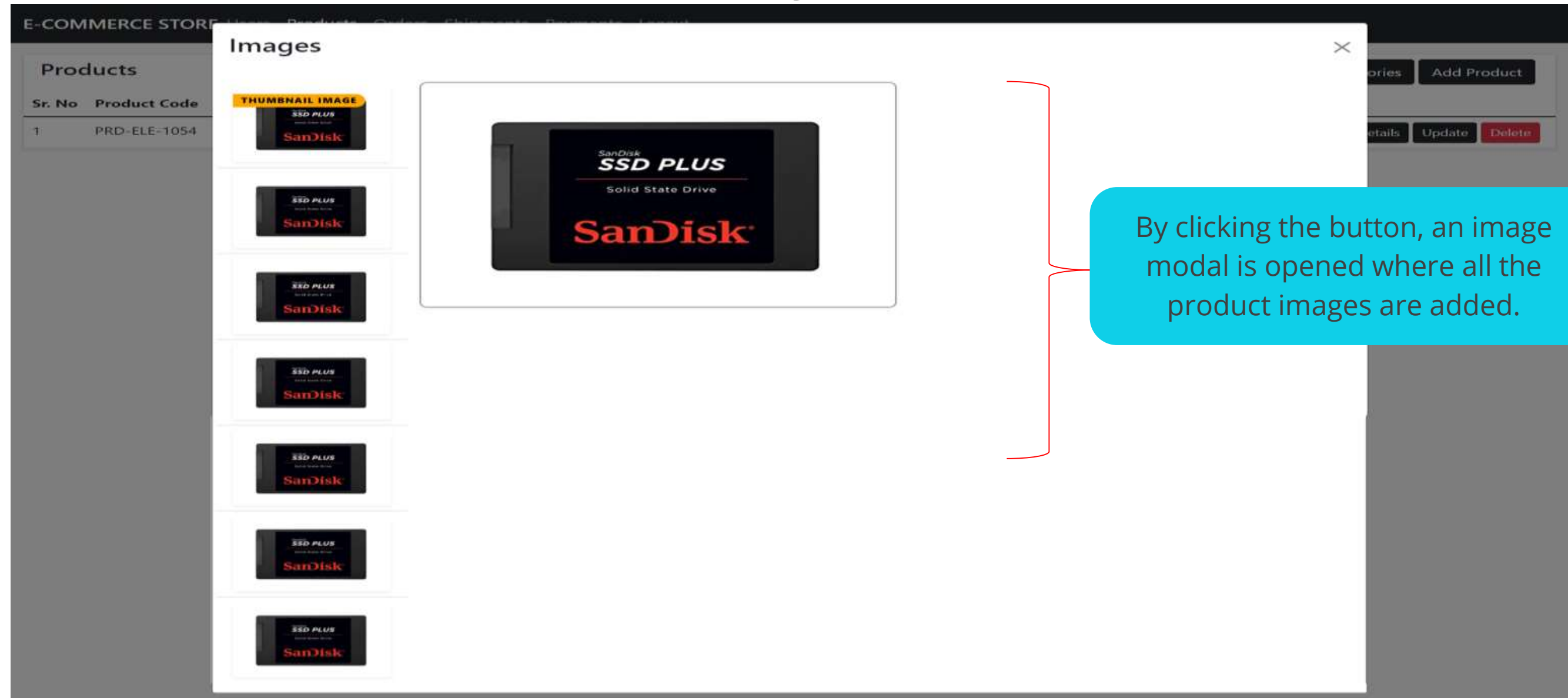
Navigation links in the sidebar include 'Manage Product Categories', 'Add Product', 'View Images', 'View Details', 'Update', and 'Delete'.

By clicking the product detail button, a dialog view is opened with the details of the product.



# Web Page for Product Component: Product Images Modal

A UI element should be added as a button in the product component to view the product images.



# Web Page for Orders Component: View Orders

In the end-user web app, users should place orders from their account. The same list of orders should be displayed in the admin panel for various users.

**E-COMMERCE STORE** [Users](#) [Products](#) [Orders](#) [Shipments](#) [Payments](#) [Logout](#)

### Orders

Placed Accepted Delivered Cancelled

Sr. No	Order Id	Order Placed	Name	Email	Contact	Total Amount	Actions
1	3456789345	03 December 2021	John Watson	john@example.com	9874563211	₹1,300	<span>View Order Details</span> <span>Update Status</span> <span>Delete</span>

In this form, the order status can be modified using the update status button.

# Web Page for Orders Component: View Orders

In the directory: src/app/pages/orders/

Command	Use
orders.component.css	CSS goes here to design the page and forms.
orders.component.html	HTML code is written in this template file.
orders.component.ts	Logic will be written in the TypeScript file.

# Web Page for Orders Component: View Products in an Order

An action button on the orders page should be View Order Details.

The screenshot shows an 'E-COMMERCE STORE' interface with a navigation bar containing 'Users', 'Products', 'Orders', 'Shipments', 'Payments', and 'Logout'. The 'Orders' section is active, displaying a table of orders. The first order is highlighted, and a 'View Order Details' button is circled in red. An arrow points from this button to a blue callout box. To the right, a 'Products' modal is open, showing a list of products with columns for 'Name / Price', 'Qty.', and 'Total Price'. The first product is 'SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s' with a quantity of 4 and a total price of ₹1,200. The modal also shows a total price of ₹300 at the bottom.

Sr. No	Order Id	Order Placed	Name	Email	Contact	Total Amount	Actions
1	3456789345	03 December 2021	John Watson	john@example.com	9874563211	₹1,300	<a href="#">View Order Details</a> <a href="#">Update Status</a> <a href="#">Delete</a>

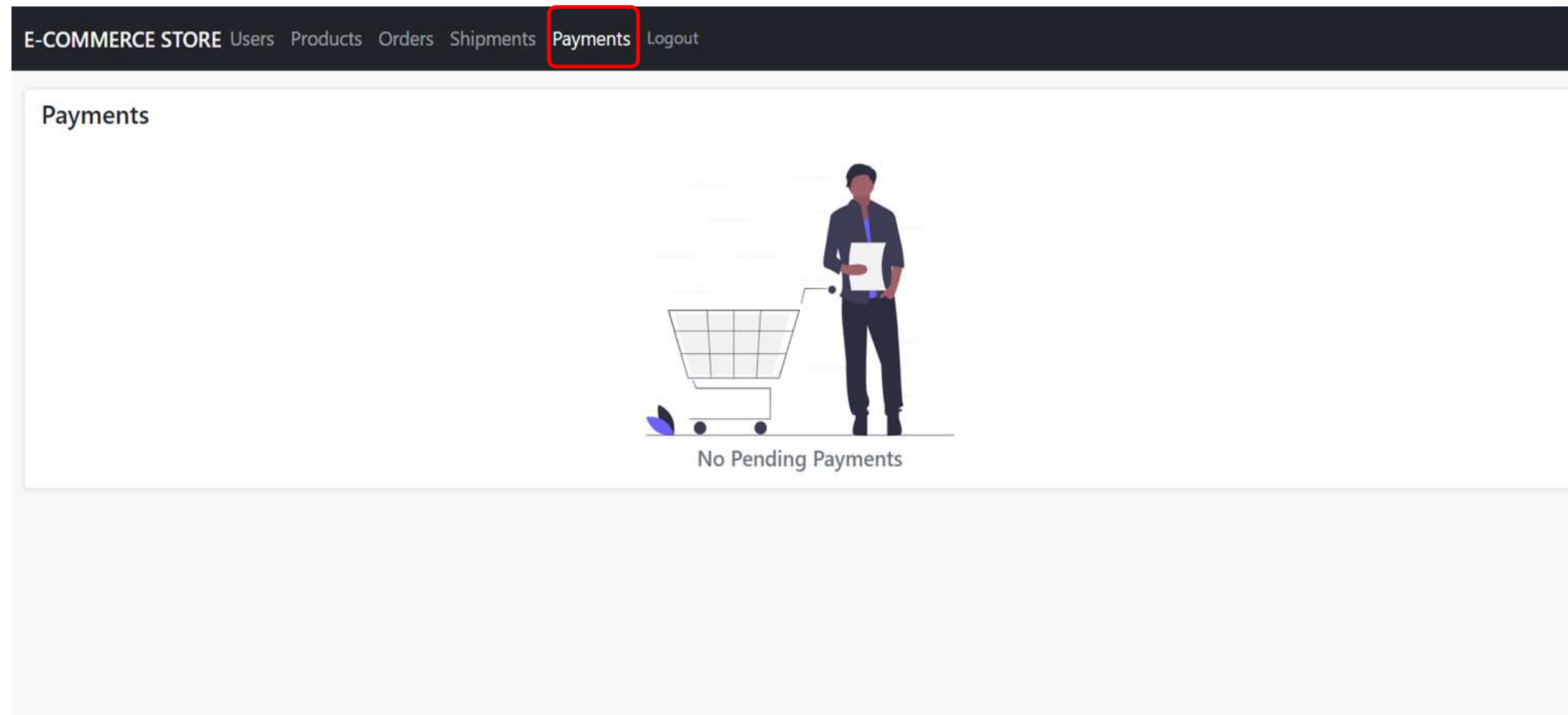
Name / Price	Qty.	Total Price
SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 Gb/s	4	₹1,200
		₹300

By clicking it, the details of an order, including the product list and total amount, are displayed.



# Web Page for Payments Component

Develop a payments page to check the payment methods and status of the payments or transactions made by the end users for the orders.



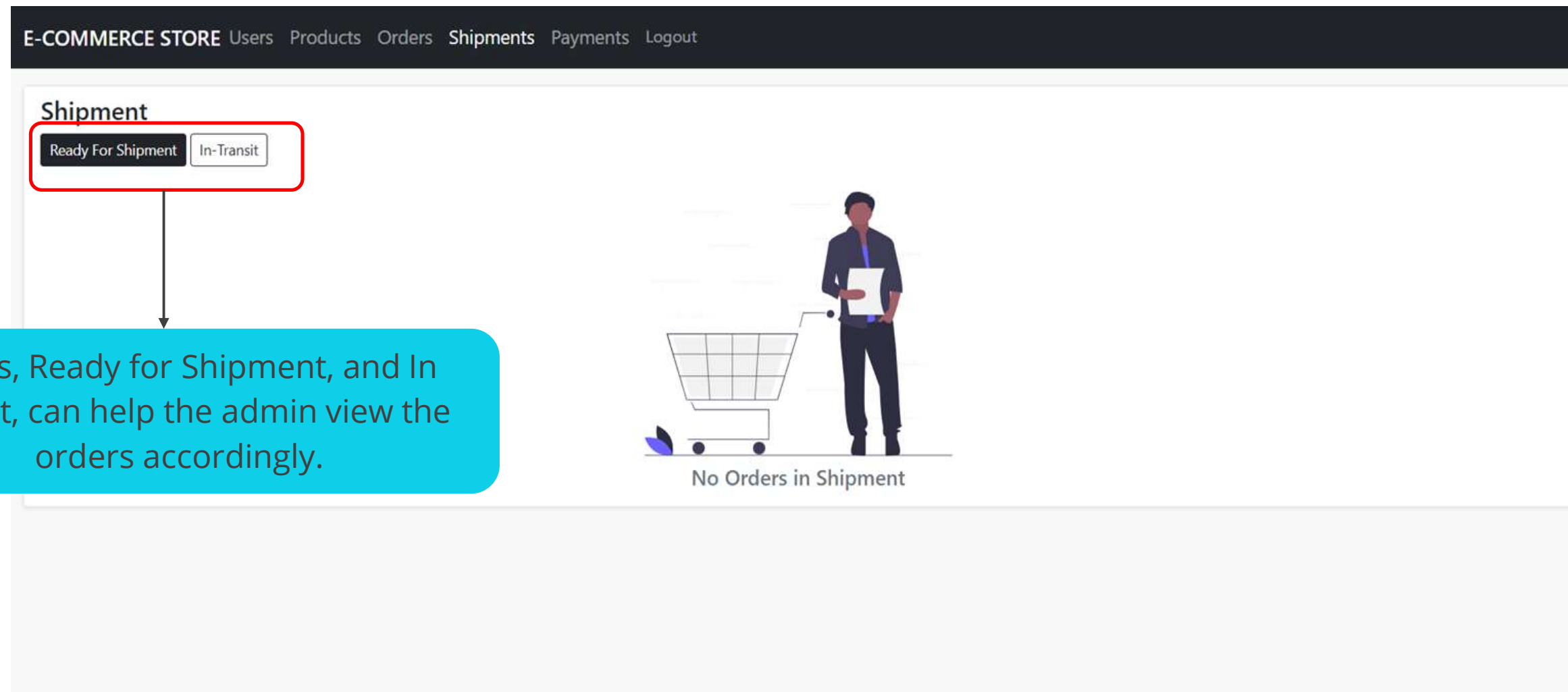
# Web Page for Payments Component

In the directory: `src/app/pages/payments/`

Command	Use
<code>payments.component.css</code>	CSS goes here to design the page and forms.
<code>payments.component.html</code>	HTML code is written in this template file.
<code>payments.component.ts</code>	Logic will be written in the TypeScript file.

# Web Page for Shipment Component

The shipments page should help the admin track the orders ready for shipment and in transit.



# Web Page for Shipment Component

In the directory: src/app/pages/shipments/

Command	Use
shipments.component.css	CSS goes here to design the page and forms.
shipments.component.html	HTML code is written in this template file.
shipments.component.ts	Logic will be written in the TypeScript file.

## Database Structure for Admin



# Creating Database in MySQL

In MySQL, the CLI uses these commands to create and work with the database.

**CREATE DATABASE ecommerce**

This command creates the e-commerce database.

**USE DATABASE ecommerce**

This command sets the current working database to e-commerce.

**SHOW TABLES**

This command lists all the tables in the database. It will return an empty set if no tables are available.

## Creating Tables

# Prerequisites

- Make sure the database is set to e-commerce.
- User can execute the following command to select the e-commerce database:

```
use database ecommerce;
```



# Creating Admin's Table for Login

To create a table for the admin, use columns such as admin ID, email, and password.

```
CREATE TABLE ADMINS
  adminId          INTEGER NOT NULL PRIMARY KEY
  AUTO_INCREMENT,
  email            VARCHAR(50) NOT NULL,
  password         VARCHAR(50) NOT NULL,
  fullName         VARCHAR(255) NOT NULL,
  loginType        INTEGER DEFAULT 1,
  addedOn          DATETIME DEFAULT CURRENT_TIMESTAMP
```



# Creating Users Table

To create a table for the user, use columns such as user ID, email, and password.

```
CREATE TABLE USERS (  
  userId          INTEGER NOT NULL PRIMARY KEY  
  AUTO_INCREMENT,  
  email           VARCHAR(50) NOT NULL,  
  password        VARCHAR(50) NOT NULL,  
  fullName        VARCHAR(255) NOT NULL,  
  street          VARCHAR(50) DEFAULT NULL,  
  city            VARCHAR(50) DEFAULT NULL,  
  state           VARCHAR(50) DEFAULT NULL,  
  country         VARCHAR(50) DEFAULT NULL,  
  pincode         INTEGER,  
  image           VARCHAR(1000),  
  contact         BIGINT,  
  addedOn         DATETIME DEFAULT CURRENT_TIMESTAMP
```





# Creating Product Categories Table

To create a table for categories, use columns such as category ID and category name.

```
CREATE TABLE CATEGORIES (  
  categoryId      INTEGER NOT NULL PRIMARY KEY  
  AUTO_INCREMENT,  
  categoryName    VARCHAR(255) NOT NULL,  
  categoryDescription VARCHAR(255),  
  categoryImageUrl VARCHAR(500),  
  active          INTEGER DEFAULT 0,  
  addedOn         DATETIME DEFAULT CURRENT_TIMESTAMP  
);
```



# Creating Products' Table

To create a table for the product, use columns such as product ID and product title.

```
CREATE TABLE PRODUCTS (  
  productId          INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,  
  productTitle       VARCHAR(500) NOT NULL,  
  productDescription VARCHAR(500) NOT NULL,  
  productCode        VARCHAR(500) NOT NULL,  
  categoryId          INTEGER,  
  images              VARCHAR(1000),  
  thumbnailImage      INTEGER DEFAULT 0,  
  price              INTEGER DEFAULT 0,  
  addedOn             DATETIME DEFAULT CURRENT_TIMESTAMP,  
  rating              INTEGER NOT NULL,  
  FOREIGN KEY (categoryId) REFERENCES CATEGORIES(categoryId)  
);
```

# Creating Orders' Table

To create a table for orders, use columns such as order ID and order date.

```
CREATE TABLE ORDERS (  
  orderId          INTEGER NOT NULL PRIMARY KEY,  
  orderDate        DATETIME DEFAULT CURRENT_TIMESTAMP,  
  orderStatus      VARCHAR(50) NOT NULL,  
  totalItems       INTEGER NOT NULL,  
  itemsSubTotal    INTEGER NOT NULL,  
  shipmentCharges  INTEGER NOT NULL,  
  totalAmount      INTEGER NOT NULL,  
  paymentStatus    INTEGER DEFAULT 0,  
  paymentStatusTitle VARCHAR(255),  
  paymentMethod     INTEGER,  
  paymentMethodTitle VARCHAR(255) NOT NULL,  
  userId           INTEGER NOT NULL,  
  name             VARCHAR(255) NOT NULL,  
  email            VARCHAR(255) NOT NULL,  
  contact          BIGINT NOT NULL,  
  address          VARCHAR(500) NOT NULL,  
  FOREIGN KEY (userId) REFERENCES USERS(userId)  
);
```



# Creating Shipments' Table

To create a table for shipments, use columns such as shipment ID and order ID.

```
CREATE TABLE SHIPMENTS(  
  shipmentId      INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,  
  orderId         INTEGER,  
  shipmentStatus  INTEGER,  
  shipmentTitle   VARCHAR(255),  
  shipmentDate    DATETIME DEFAULT CURRENT_TIMESTAMP,  
  expectedDeliveryDate DATETIME,  
  shipmentMethod  VARCHAR(255),  
  shipmentCompany VARCHAR(255),  
  FOREIGN KEY (orderId) REFERENCES ORDERS(orderId)  
);
```

# Creating Order Items' Table

To create a table for order items, use columns such as order item ID and order ID.

```
CREATE TABLE ORDERITEMS (  
  orderItemId      INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,  
  orderId          INTEGER,  
  productId        INTEGER,  
  productCode      VARCHAR(255) NOT NULL,  
  productImg       VARCHAR(255) NOT NULL,  
  productTitle     VARCHAR(255) NOT NULL,  
  productDescription VARCHAR(255) NOT NULL,  
  productCategory  VARCHAR(255) NOT NULL,  
  price            INTEGER NOT NULL,  
  quantity         INTEGER NOT NULL,  
  totalPrice       INTEGER NOT NULL,  
  FOREIGN KEY (orderId) REFERENCES ORDERS (orderId),  
  FOREIGN KEY (productId) REFERENCES PRODUCTS (productId)  
);
```





## Key Takeaways

- HTML and CSS are used for the development of the admin dashboard.
- Angular templates are used for web page development.
- Various Angular CLI commands, such as `ng serve`, are used to view the web app.
- The UI is developed on the web page for admin users to manage various product categories.



## Key Takeaways

- The action button on the product component web page adds the product to the database.
- In the end-user web app, users can place orders from their account, and the same list of orders is displayed in the admin panel for updating order status by an admin user.
- The shipments page helps the admin track the orders that are ready for shipment and in transit.
- The database is set to e-commerce to create tables.

