TECHNOLOGY

**Project Session 03:**
**End User Web Pages and Data**
**Structure for End User**

simplilearn

# You Already Know

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



HTML



CSS



SQL

# A Day in the Life of a Full Stack Developer

As a full stack web developer, our key role is to develop both client and server software.

Angular can be used to build the front end of the web page.

Spring Boot and MySQL or MongoDB can be used to build the back end.

# A Day in the Life of a Full Stack Developer

Now, Bob needs to create web page and database for end user Web App to further proceed with his project.

Let us use previously discussed technologies to create the web page and database for the end user web app.

In this lesson, we will learn HTML and CSS skills to design and develop web pages in the previously created Angular project. We will also use SQL commands to create tables in the database for the end-user web app, helping 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 end-user web app

◉ Develop the CSS for styling the web pages and configure routing

◉ Use SQL commands to create tables for the end-user web app

◉ Create primary and foreign key relationships within tables for data integrity

Angular Component Templates for End User

# Modules

In this project, various modules will be managed, such as:

Listing the products for end user

Managing the cart

Managing the Whishlist and orders

# Web Page Templates

For basic operations, the web page templates are developed, using:

# Web Page for Authentication Login Component

Develop a login page for the end user to sign in and enter the home page.

E-COMMERCE STORE
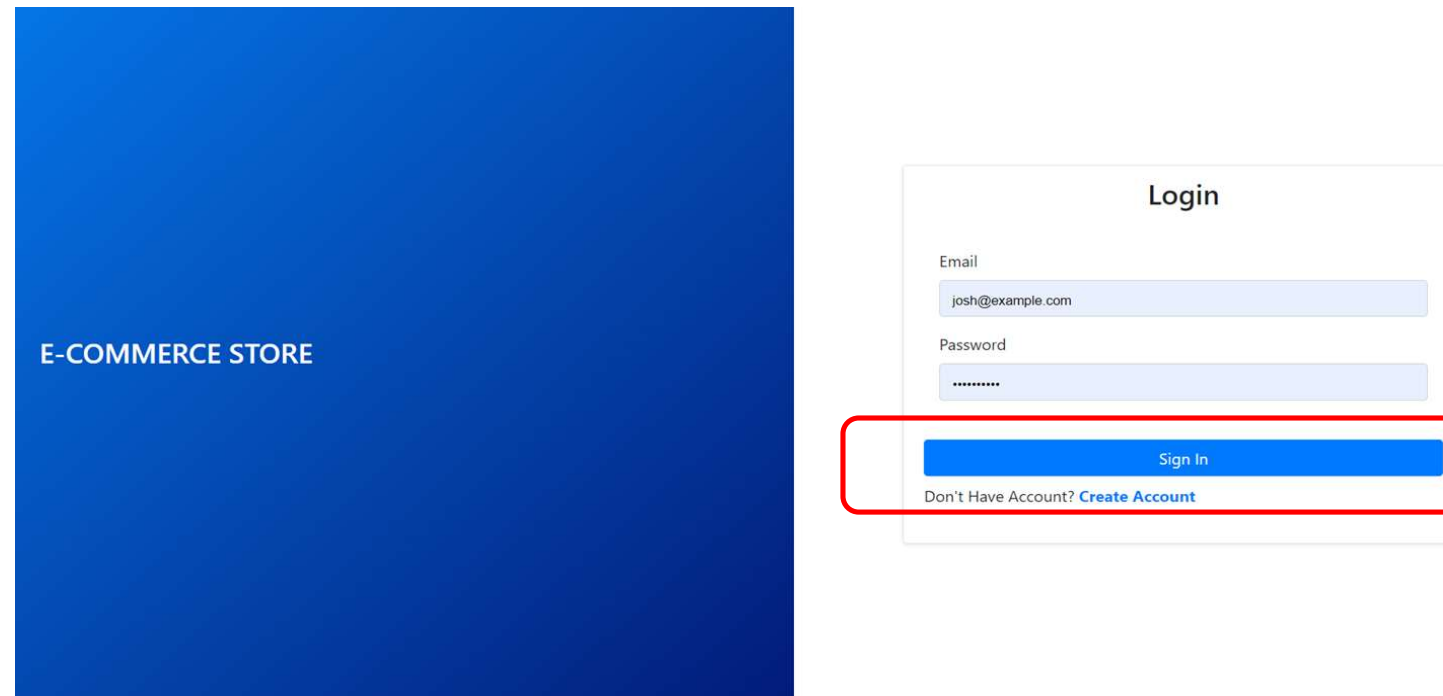
Login page

# Web Page for Authentication Login Component

Design the login page with the email, password, and a login button.



Click on the login button to validate the user and navigate to the home page, which will have a navigation bar with various options.

# Web Page for Authentication Login Component

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

**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 User Registration Component

A new user needs to register.



**E-COMMERCE STORE**

**Register**

Name
eg: John Peterson

Email
eg: abc@example.com

Password
********

Sign Up

Already Have Account? **Sign In**

# Web Page for User Registration Component

Design the register component to take the name, email, and password as input to register the user on the e-store web app.
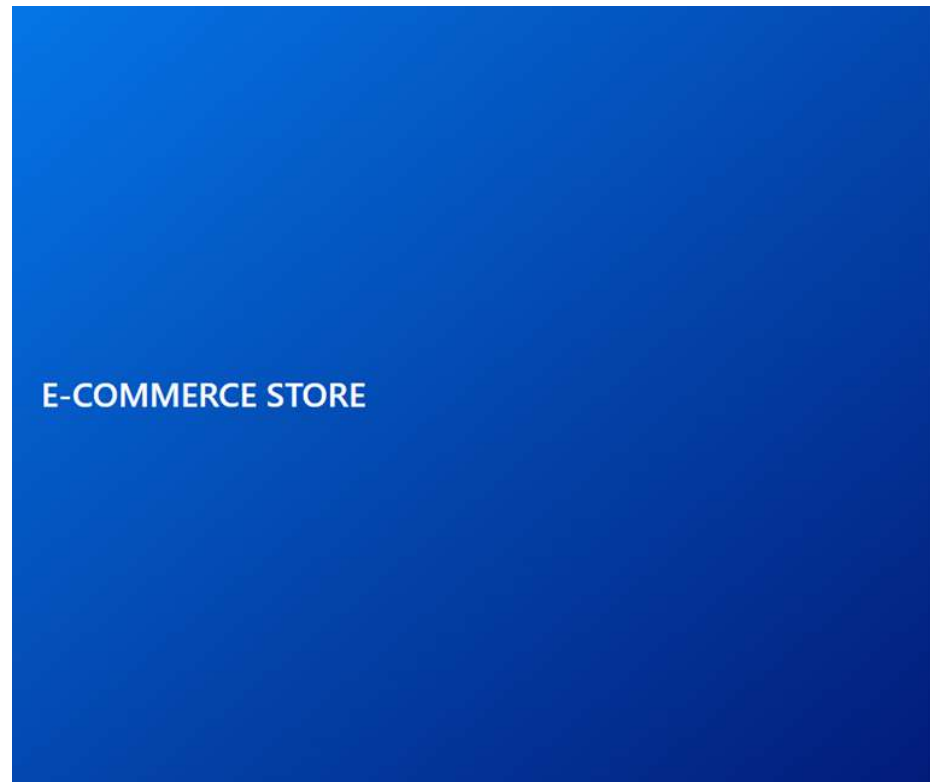


E-COMMERCE STORE

Register

Name
eg: John Peterson

Email
eg: abc@example.com

Password
********

Sign Up

Already Have Account? Sign In

After registration, navigate to the home page which will have a navigation bar with various options.

# Web Page for User Registration Component

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

**register.component.css**

CSS goes here to design the page and forms.

**register.component.html**

HTML code is written in this template file.

**register.component.ts**

Logic will be written in typescript file.
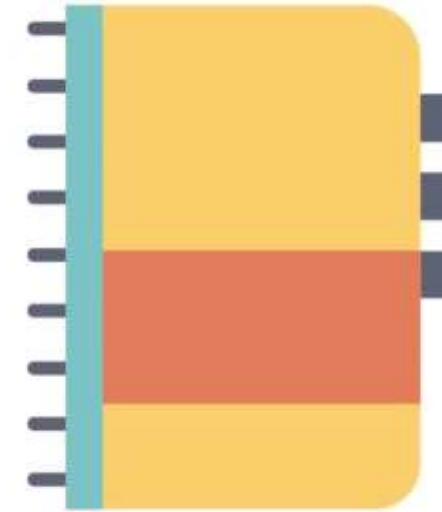
# Web Page for Profile Component

User should be able to manage their profile by navigating to the profile component on navigation bar.

# Web Page for Profile Component

The web page should have details of the user. The user should be able to manage contact information, which includes the address for delivering the order.

# Web Page for Profile Component

In the directory: src/app/pages/user-profile/

**user-profile.component.css**

CSS goes here to design the page and forms.

**user-profile.component.html**

HTML code is written in this template file.

**user-profile.component.ts**

Logic will be written in typescript file.

# Web Page for Products Component

UI should be developed to show the various products added by the admin.
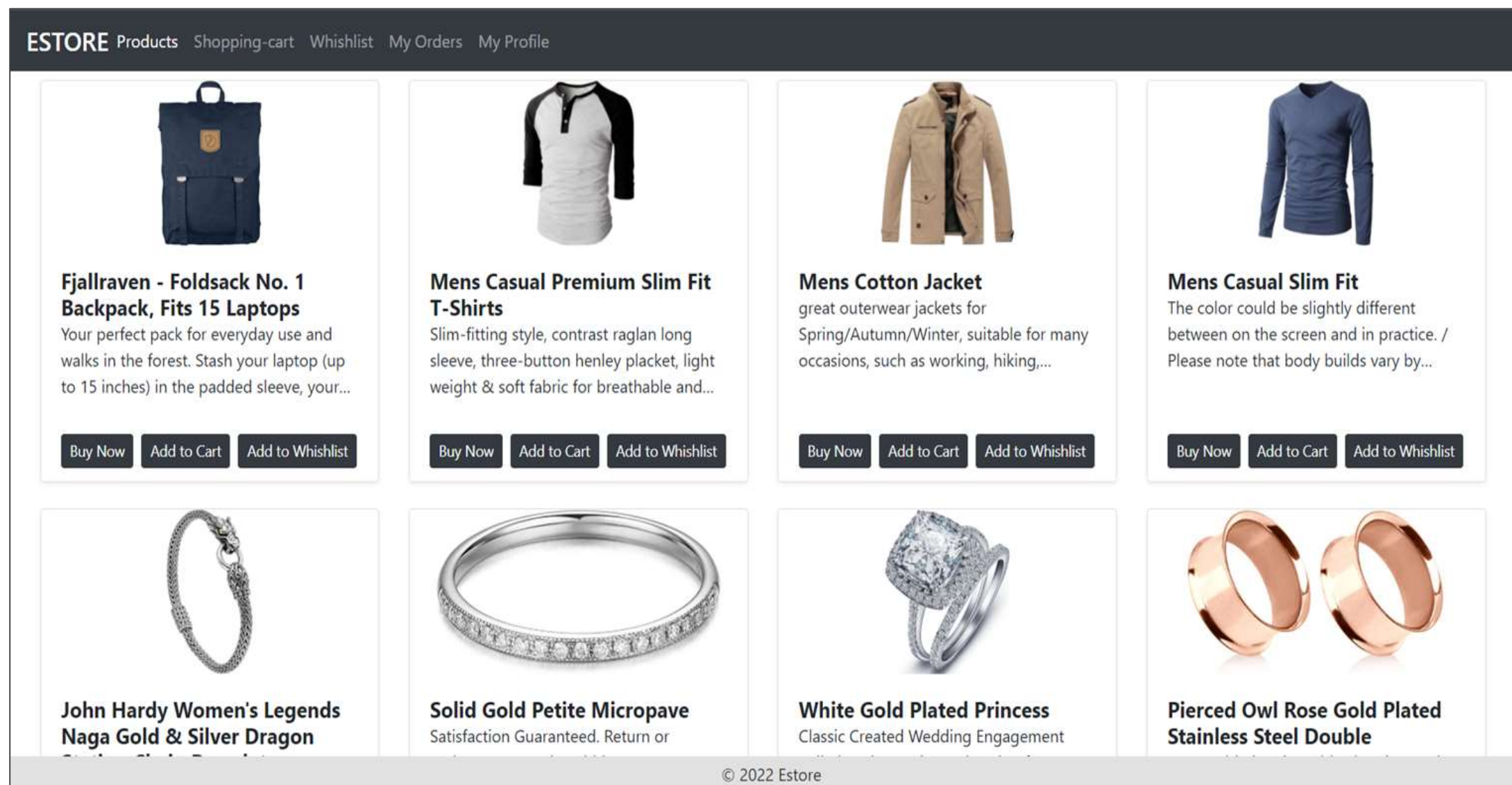
# Web Page for Products Component

All the products should be listed with options to buy the product, add it to the cart, or add it to the Whishlist.

# Web Page for Products Component

Search, sort, and filter operations can also be implemented on the same web page.
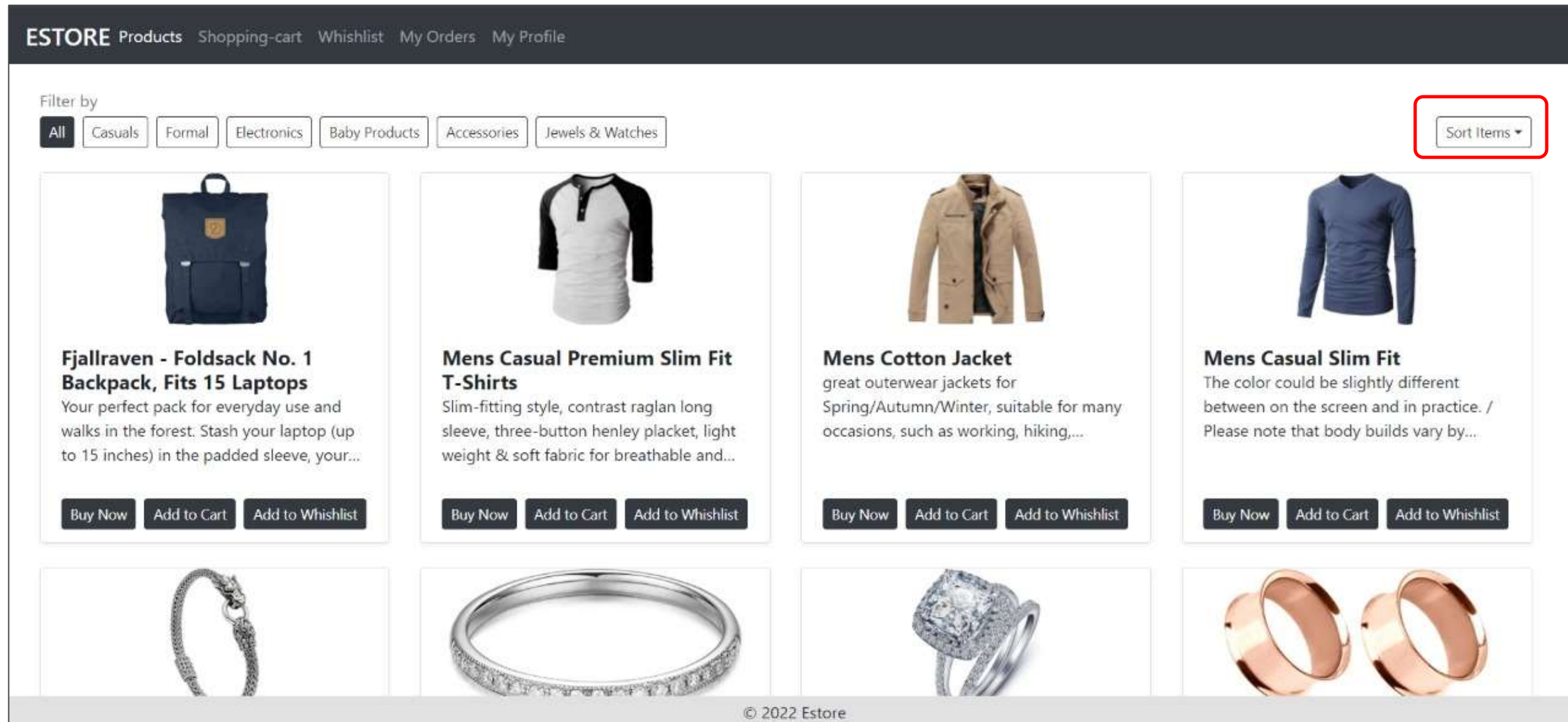
# Web Page for Products Component

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

**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.

# Web Page for Whishlist Component

User should be able to mark any product as favorite by moving it to the Whishlist.

# Web Page for Whishlist Component

The Whishlist component should show the list of products that a user may buy in the future.

# Web Page for Whishlist Component

There should be a button to move the product to cart by clicking the same appearing on the product item.

# Web Page for Whishlist Component

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

**Whishlist.component.css**

CSS goes here to design the page and forms.

**Whishlist.component.html**

HTML code is written in this template file.

**Whishlist.component.ts**

Logic will be written in typescript file.

# Web Page for Cart Component

The navigation bar is used to navigate to the cart tab, where users should see a list of the products along with their quantities and prices.

# Web Page for Cart Component

A summary of the products to be placed should be shown along with the option to buy, which is to place an order.

# Web Page for Cart Component

Payment's method page can also be implemented.

# Web Page for Cart Component

In the directory: src/app/pages/shopping-cart/

**shopping-cart.component.css**

CSS goes here to design the page and forms.

**shopping-cart.component.html**

HTML code is written in this template file.

**Shopping-cart.component.ts**

Logic will be written in typescript file.

# Web Page for Checkout Component

From the Cart, user should navigate to the final checkout page.

**ESTORE** Products  Shopping-cart **6**  Whishlist  My Orders  My Profile

## Payment Method

### Add Debit/Credit/ATM Card
You can save your Mastercard and VISA cards as per new RBI guidelines

VISA  Mastercard  RuPay

### Netbanking
For faster payment and instant refund, please use UPI

### Other UPI

### Pay on Delivery
Pay digitally with SMS Pay Link. Cash may not be accepted in COVID restricted areas.

## Order Summary

Order Total  ₹1,251.46

## Shipping Address

Street
**16, 1st Floor., Agboat Wala Bldg., Abdulrehman Street No 282, Nr Islami Hotel, Abdul Rehman Street**

City
**Mumbai**

State
**Maharashtra**

Postal Code
**400003**

**Place order**

© 2022 Estore

# Web Page for Checkout Component

User should choose the payment method and address for the confirmation.

ESTORE  Products  Shopping-cart **6**  Whishlist  My Orders  My Profile

## Payment Method

**Add Debit/Credit/ATM Card**
You can save your Mastercard and VISA cards as per new RBI guidelines

VISA  mastercard  RuPay»

**Netbanking**
For faster payment and instant refund, please use UPI

**Other UPI**

**Pay on Delivery**
Pay digitally with SMS Pay Link. Cash may not be accepted in COVID restricted areas.

## Order Summary

Order Total                    ₹1,251.46

## Shipping Address

Street
**16, 1st Floor., Agboat Wala Bldg., Abdulrehman Street No 282, Nr Islami Hotel, Abdul Rehman Street**

City
**Mumbai**

State
**Maharashtra**

Postal Code
**400003**

Place order

© 2022 Estore

# Web Page for Checkout Component

Finally, click on the place order button, and a successful order creation message will be shown to the user.

# Web Page for Checkout Component

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

**checkout.component.css**

CSS goes here to design the page and forms.

**checkout.component.html**

HTML code is written in this template file.

**checkout.component.ts**

Logic will be written in typescript file.

# Web Page for Orders Component: View Orders

As the user places an order from their account, the list of orders should be displayed to them for reference purposes with order status.

**ESTORE** Products  Shopping-cart  Whishlist  **My Orders**  My Profile

| Order Placed | Total | Ship To | | Order ID: **#3456789345** |
|---|---|---|---|---|
| **03 December 2021** | **₹839.60** | **John Watson** | | View Order Details |

**John Hardy Women's Legends Naga Gold & Silver Dragon Station Chain Bracelet**
From our Legends Collection, the Naga was inspired by the mythical water dragon that protects the ocean's pearl. Wear facing inward to be bestowed with love and abundance, or outward for protection.

₹695.00

**Mens Casual Premium Slim Fit T-Shirts**
Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.

₹22.30

© 2022 Estore

simplilearn

# Web Page for Orders Component: View Orders

In the future, the detailed model should be opened when connected to the backend with the click of View order details.



**ESTORE** Products  Shopping-cart  Whishlist  **My Orders**  My Profile

| Order Placed | Total | Ship To | |
|---|---|---|---|
| **03 December 2021** | **₹839.60** | **John Watson** | Order ID: **#3456789345** <br> View Order Details |

**John Hardy Women's Legends Naga Gold & Silver Dragon Station Chain Bracelet**
From our Legends Collection, the Naga was inspired by the mythical water dragon that protects the ocean's pearl. Wear facing inward to be bestowed with love and abundance, or outward for protection.
₹695.00

**Mens Casual Premium Slim Fit T-Shirts**
Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.
₹22.30

© 2022 Estore

# Web Page for Orders Component - View Orders

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

**order.component.css**

CSS goes here to design the page and forms.

**order.component.html**

HTML code is written in this template file.

**order.component.ts**

Logic will be written in typescript file.

**BUY**

TECHNOLOGY

**Database Structure for End User**

# Creating Database in MySQL

MySQL CLI usage to create and work with database

**Use database e-commerce**
Selects current working database

**02**

**Create database e-commerce**
Creates the database for e-commerce

**01**

**Show tables**
Lists all the tables in the database

**03**
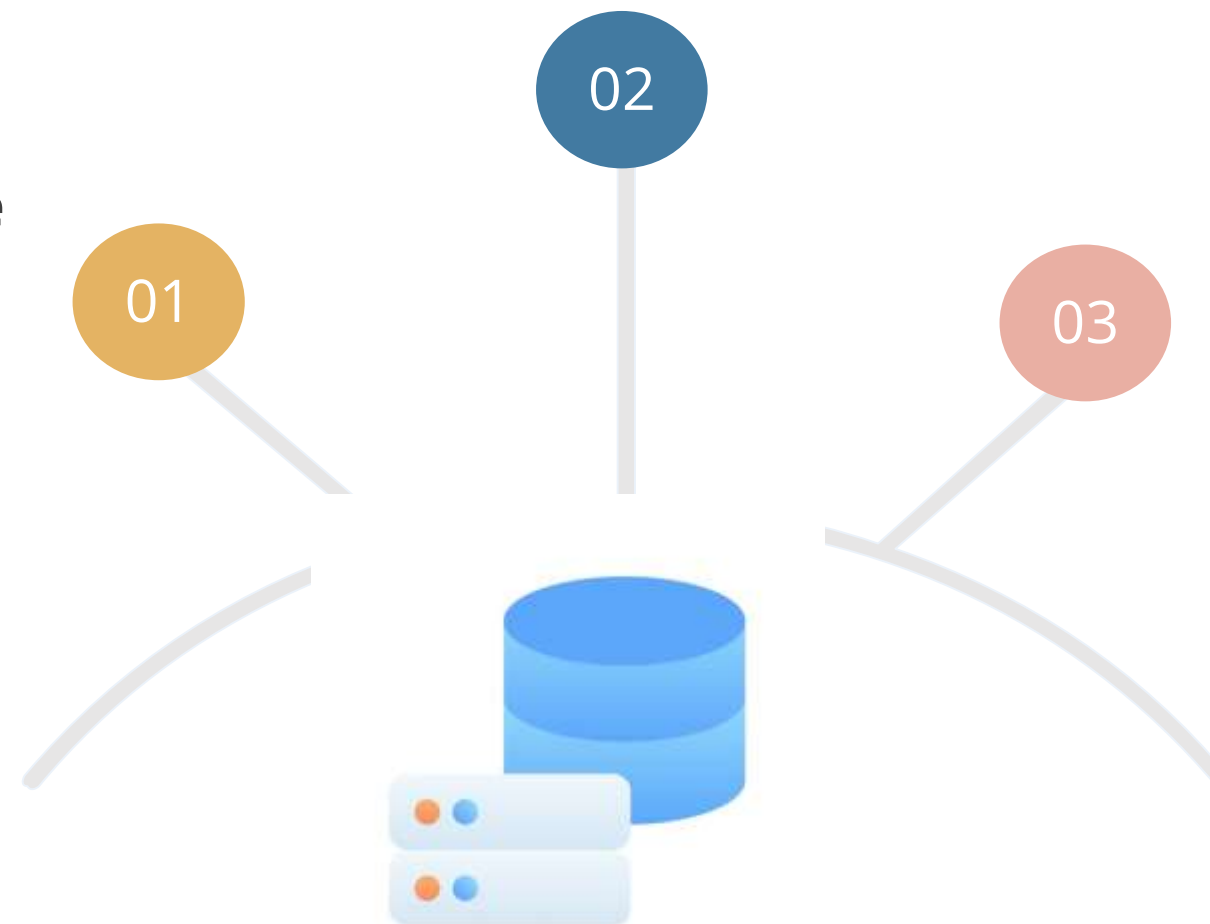
# Creating Tables

# Creating Table

Many tables are common for the dashboards.
Example: Order Table in Admin will be used by the Web App for end users as well.

Only the Tables required for Cart and the Whishlist should be created.
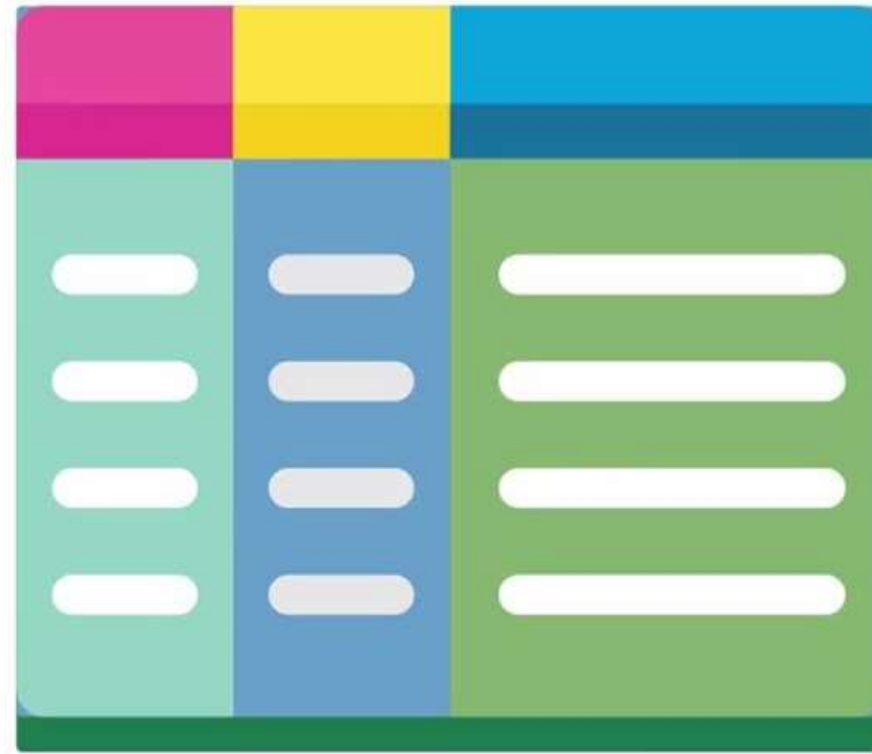
The rest of the tables are already created during the Admin table structure.

# Generating User Table for Login

Ensure the database is selected as ecommerce, or you can execute the command to choose the ecommerce database as follows:

**Use database ecommerce;**

```
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
);
) NOT NULL,
```

# Generating User Table for Login

For the simplicity, one address per user is considered.

In case one user has many addresses, then Address tables are created separately with primary and foreign key relationships.

# Generating Whishlist Table

Make sure the database is selected as ecommerce, or you can execute the following command to select the ecommerce database:

**Use database ecommerce;**

```
CREATE TABLE WHISHLIST {

  WhishlistId           INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,

  productId           INTEGER NOT NULL,

  userId              INTEGER NOT NULL,

  FOREIGN KEY (productId) REFERENCES PRODUCTS(productId),

  FOREIGN KEY (userId) REFERENCES USERS(userId)

}
```

# Generating Cart Table

Make sure the database is selected as ecommerce, or you can execute the following command to make the database selection for ecommerce as :

**Use database ecommerce;**

```
CREATE TABLE CART {

  cartId              INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,

  productId           INTEGER NOT NULL,

  quantity            INTEGER DEFAULT 1,

  userId              INTEGER NOT NULL,

  FOREIGN KEY (productId) REFERENCES PRODUCTS(productId),

  FOREIGN KEY (userId) REFERENCES USERS(userId)

}
```

# Key Takeaways

- For basic operations, the web page templates are developed using HTML and CSS.

- User can manage their profile by navigating to the profile component on the navigation bar.

- UI is developed to show the various products added by the admin.

- The Whishlist component shows the list of products that a user may buy in the future.

simplilearn

# Key Takeaways

- The navigation bar is used to navigate to the cart tab, where the user can see a list of the products along with their quantity and price.

- MySQL CLI  is used  to create and work with database.

- Address table are created separately with primary and foreign key relationship.

simplilearn