

1 Create Product List

1.0 Set Up the Product List file

1) Create the component of the product list

The screenshot shows the StackBlitz IDE interface. On the left, the project structure is visible with a 'src' folder containing an 'app' folder. A context menu is open over the 'app' folder, showing options like 'Angular Generator', 'Rename', and 'Delete'. The 'Component' option is selected. In the center, the code editor shows a partial HTML file named 'product-list.component.html' with the following content:

```
<h2>Products</h2>


<h3>
  [title]="product.name + ' details'"
  [routerLink]="/products", product.id
</h3>
<a href="#">
  {{ product.name }}
</a>


```

On the right, the preview window shows a web application titled 'My Store' with a 'Products' section. It lists two phones: 'Phone XL' and 'Phone Mini', each with a 'Share' button. Below the phones, a message says 'product-alert works!'. The bottom right corner of the preview shows the URL <https://angular-7vcf.stackblitz.io>.

2) Add `*ngFor` statement to the following content so that the product object in the `product.ts` file can be gone through and shown on the web page

Angular (forked) - StackBlitz

stackblitz.com/edit/angular-7vctw2-wzpnrm?file=src%2Fapp%2Fproduct-li...

应用 百度一下，你就知道 rit email 有道首页 Employee-facing re...

Save Fork Share Angular (forked) Open in New Window LIVE Close

INFO FILES

product-list.component.html

```
<h2>Products</h2>
<div *ngFor="let product of products">
  <h3>
    <a [title]="product.name + ' details'" [routerLink]=["'/products', product.id]">
      {{ product.name }}
    </a>
  </h3>
  <p *ngIf="product.description">Description: {{ product.description }}</p>
  <button (click)="share()">Share</button>
  <app-product-alert [product]="product" (notify)="onNotify()">
    </app-product-alert>
  </div>
<!--
Copyright Google LLC. All Rights Reserved.
Use of this source code is governed by an MIT-style
license that
can be found in the LICENSE file at
https://angular.io/license
-->
```

My Store

Phone XL

Description: A large phone with one of the best screens

Share

product-alert works!

Notify Me

Phone Mini

Description: A great phone with one of the best cameras

Share

product-alert works!

Phone Standard

Share

Angular (forked)

product-list.component.html

```
<h2>Products</h2>
<div *ngFor="let product of products">
  <h3>
    <a [title]="product.name + ' details'" [routerLink]=["'/products', product.id]">
      {{ product.name }}
    </a>
  </h3>
  <p *ngIf="product.description">Description: {{ product.description }}</p>
  <button (click)="share()">Share</button>
  <app-product-alert [product]="product" (notify)="onNotify()">
    </app-product-alert>
  </div>
<!--
Copyright Google LLC. All Rights Reserved.
Use of this source code is governed by an MIT-style
license that
can be found in the LICENSE file at
https://angular.io/license
-->
```

My Store

Checkout

Products

Phone XL

Description: A large phone with one of the best screens

Share

product-alert works!

Notify Me

Phone Mini

Description: A great phone with one of the best cameras

Share

product-alert works!

Phone Standard

Share

3) Add the following statement so that it will show the information of title when the mouse points to the product name.

```

1 <h2>Products</h2>
2
3 <div *ngFor="let product of products">
4   <h3>
5     <a
6       [title]="'product.name + ' details'"
7       [routerLink]="/products", product.id]
8     >
9     | {{ product.name }}
10    | </a>
11   </h3>
12   <p *ngIf="product.description">Description: {{ product.description }}</p>
13   <button (click)="share()">Share</button>
14   <app-product-alert [product]="product" (notify)="onNotify()">
15   </app-product-alert>
16 </div>
17 <!--
18 Copyright Google LLC. All Rights Reserved.
19 Use of this source code is governed by an MIT-style license that
20 can be found in the LICENSE file at https://angular.io/license
21 -->

```

4) Add the *ngIf statement so that the object of the product.ts will be gone through. If the object has description variable, the statement of the html file will be executed and the description information will be shown on the product list web page.

```

1 <h2>Products</h2>
2
3 <div *ngFor="let product of products">
4   <h3>
5     <a
6       [title]="'product.name + ' details'"
7       [routerLink]="/products", product.id]
8     >
9     | {{ product.name }}
10    | </a>
11   </h3>
12   <p *ngIf="product.description">Description: {{ product.description }}</p>
13   <button (click)="share()">Share</button>
14   <app-product-alert [product]="product" (notify)="onNotify()">
15   </app-product-alert>
16 </div>
17 <!--
18 Copyright Google LLC. All Rights Reserved.
19 Use of this source code is governed by an MIT-style license that
20 can be found in the LICENSE file at https://angular.io/license
21 -->

```

5) Add the button tag to the html file. This file will call on the share() function which defined on the product-list.component.ts file.

product-list.component.html

```

1 <h2>Products</h2>
2
3 <div *ngFor="let product of products">
4   <h3>
5     <a [title]="product.name + ' details'" 
6       [routerLink]="/products", product.id)>
7       {{ product.name }}
8     </a>
9   </h3>
10  <p *ngIf="product.description">Description: {{ product.description }}</p>
11  <button (click)="share()". Share</button>
12  <app-product-alert [product]="product" (notify)="onNotify()".>
13  </app-product-alert>
14
15
16 </div>
17 <!--
18 Copyright Google LLC. All Rights Reserved.
19 Use of this source code is governed by an MIT-style license that
20 can be found in the LICENSE file at https://angular.io/license
21 -->

```

Copyright Google LLC. All Rights Reserved.
Use of this source code is governed by an MIT-style license that
can be found in the LICENSE file at https://angular.io/license

Console LFH4Z 2125 9/18/2021

1.1 Make Use of Child Component

1) Create a product alert component and add the component to the app.module.ts file

app.module.ts

```

13 import { ShippingComponent } from
14   './shipping/shipping.component';
15 @NgModule({
16   imports: [
17     BrowserModule,
18     ReactiveFormsModule,
19     HttpClientModule,
20     RouterModule.forRoot([
21       { path: '', component: ProductListComponent },
22       { path: 'products/:productId', component:
23         ProductDetailsComponent },
24       { path: 'cart', component: CartComponent },
25       { path: 'shipping', component:
26         ShippingComponent },
27     ]),
28   ],
29   declarations: [
30     AppComponent,
31     TopBarComponent,
32     ProductListComponent,
33     ProductAlertComponent,
34     ProductDetailsComponent,
35     CartComponent,
36     ShippingComponent,
37   ],
38   bootstrap: [AppComponent],
39 }
40 Copyright Google LLC. All Rights Reserved.
41 Use of this source code is governed by an MIT-style
42 license that
43 can be found in the LICENSE file at

```

LFH4Z 21:32 9/18/2021

2) Add Input class from the @angular/core. This input object will receive the input object from the parent component which make use of this component.

```

1 import { Component, OnInit } from '@angular/core';
2 import { Input } from '@angular/core';
3 import { Output, EventEmitter } from '@angular/core';
4 import { Product } from '../products';
5
6 @Component({
7   selector: 'app-product-alert',
8   templateUrl: './product-alert.component.html',
9   styleUrls: ['./product-alert.component.css'],
10 })
11 export class ProductAlertComponent implements OnInit {
12   @Input() product!: Product;
13   @Output() notify = new EventEmitter();
14   constructor() {}
15   ngOnInit() {}
16 }

```

3) Add the product alert component to the product list web page. The [product] = "product" defines the object to be sent to the product-alert component. The (notify) = "onNotify()" means when the notify object sent from the child component is received, the parent component will call the onNotify() function.

```

1 <h2>Products</h2>
2
3 <div *ngFor="let product of products">
4   <h3>
5     <a
6       [title]="product.name + ' details'"
7       [routerLink]=["/products", product.id]"
8     >
9       {{ product.name }}
10    </a>
11  </h3>
12  <p *ngIf="product.description">Description: {{ product.description }}</p>
13  <button (click)="share()">Share</button>
14  <app-product-alert [product]="product" (notify)="onNotify()">
15  </app-product-alert>
16 </div>
17 <!--
18 Copyright Google LLC. All Rights Reserved.
19 Use of this source code is governed by an MIT-style
20 license that
21 can be found in the LICENSE file at
22 https://angular.io/license
-->

```

4) At the product-list.component.ts file, add the onNotify() function to verify the successful execution when the child component is called on.

The screenshot shows a browser window with two main sections: a code editor on the left and a live preview on the right.

Code Editor (product-list.component.ts):

```
1 import { Component } from '@angular/core';
2
3 import { products } from '../products';
4
5 @Component({
6   selector: 'app-product-list',
7   templateUrl: './product-list.component.html',
8   styleUrls: ['./product-list.component.css'],
9 })
10 export class ProductListComponent {
11   products = products;
12
13   share() {
14     window.alert('The product has been shared!');
15   }
16
17   onNotify() {
18     window.alert('You will be notified when the
19       product goes on sale');
20   }
21
22 Copyright Google LLC. All Rights Reserved.
23 Use of this source code is governed by an MIT-style
24 license that
25 can be found in the LICENSE file at
26 https://angular.io/license
27 */
```

A red box highlights the `onNotify()` method and its alert message.

Live Preview:

- My Store** - Header with a shopping cart icon and "Checkout" button.
- Products** section:
 - Phone XL**: Description: A large phone with one of the best screens. Includes a "Share" button and a "product-alert works!" message with a "Notify Me" button.
 - Phone Mini**: Description: A great phone with one of the best cameras. Includes a "Share" button and a "product-alert works!" message.
 - Phone Standard**: Includes a "Share" button.
- Bottom status bar: LFH4Z, 21:42, 9/18/2021, and a few icons.

5) Add EventEmitter to the product-alert.ts file so that the component can use the object of the EventEmitter to notify the parent component

The screenshot shows a browser window with an Angular application. The URL is <https://angular-7vctw2-wzpn..>. The application displays three products: Phone XL, Phone Mini, and Phone Standard. Each product has a title, a description, and a 'Share' button. The Phone XL card is currently active, showing its details: "Description: A large phone with one of the best screens". The Phone Mini and Phone Standard cards also have descriptions and share buttons. On the left side of the screen, the product-alert.component.ts file is open in a code editor. The code implements an OnInit interface and uses @Input() and @Output() decorators. The @Output() notify line is highlighted with a red underline.

```
product-alert.component.ts X

1 import { Component, OnInit } from '@angular/core';
2 import { Input } from '@angular/core';
3 import { Output, EventEmitter } from '@angular/core';
4 import { Product } from '../products';
5
6 @Component({
7   selector: 'app-product-alert',
8   templateUrl: './product-alert.component.html',
9   styleUrls: ['./product-alert.component.css'],
10 })
11 export class ProductAlertComponent implements OnInit {
12   @Input() product!: Product;
13   @Output() notify = new EventEmitter();
14   constructor() {}
15   ngOnInit() {}
16 }
```

My Store Checkout

Products

Phone XL

Description: A large phone with one of the best screens

Share

product-alert works!

Notify Me

Phone Mini

Description: A great phone with one of the best cameras

Share

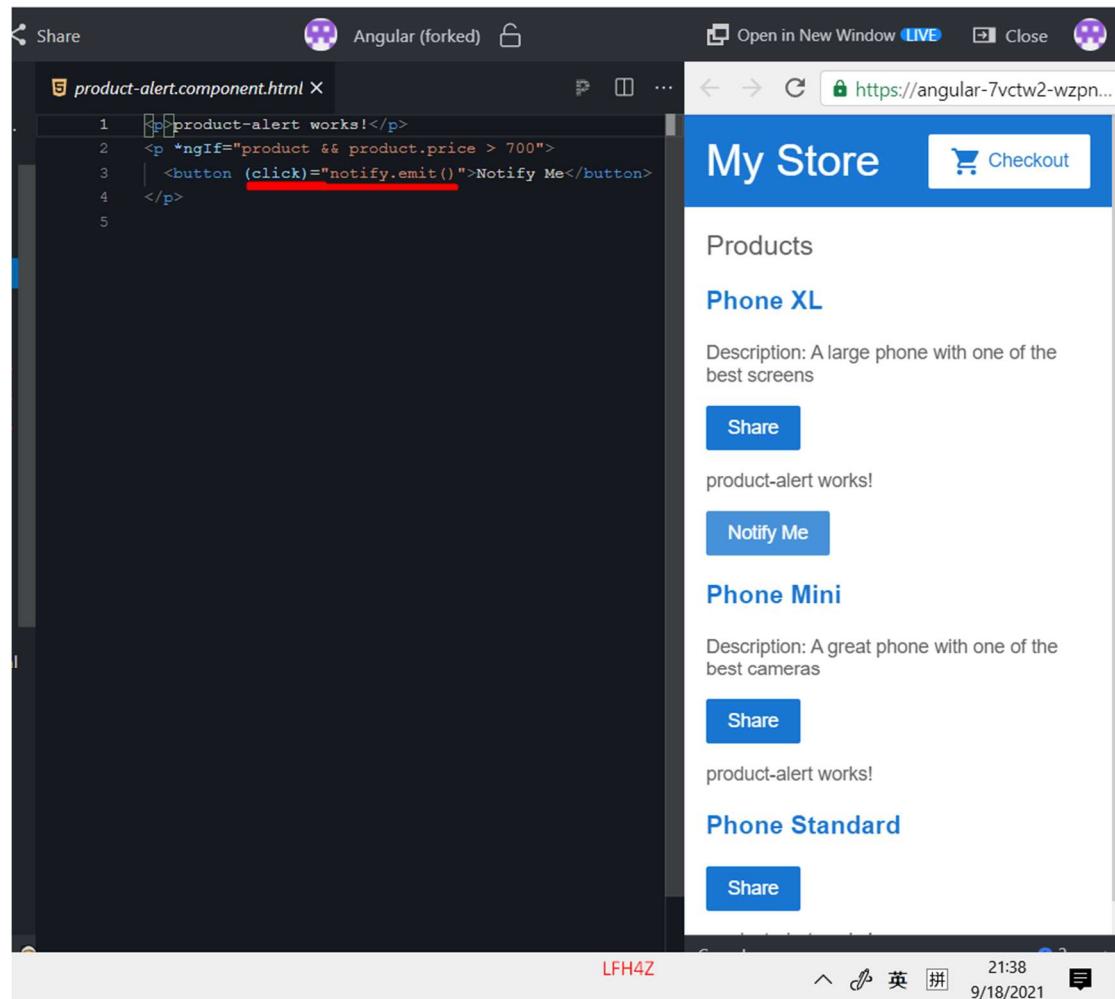
product-alert works!

Phone Standard

Share

LFH4Z 21:37 9/18/2021 ⌂ 英 拼

6) Add the content to the product alert html file. When the parent component call the child component, it will show on the webpage of the parent component.



7) Verification

angular-7vctw2-wzpnrm.stackblitz.io 上的嵌入式页面显示

You will be notified when the product goes on sale

确定

My Store

Products

Phone XL

Description: A large phone with one of the best screens

Share

product-alert works!

Notify Me

Phone Mini

Description: A great phone with one of the best cameras

Share

product-alert works!

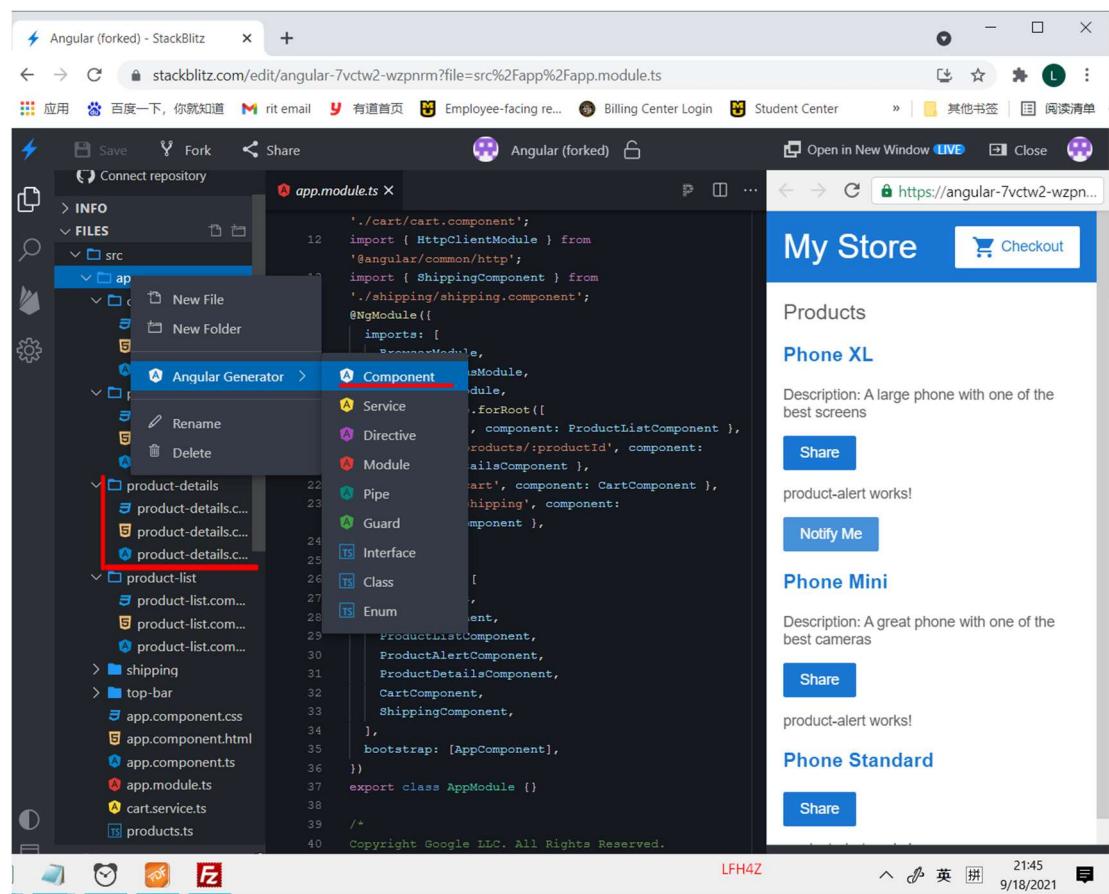
Phone Standard

Share

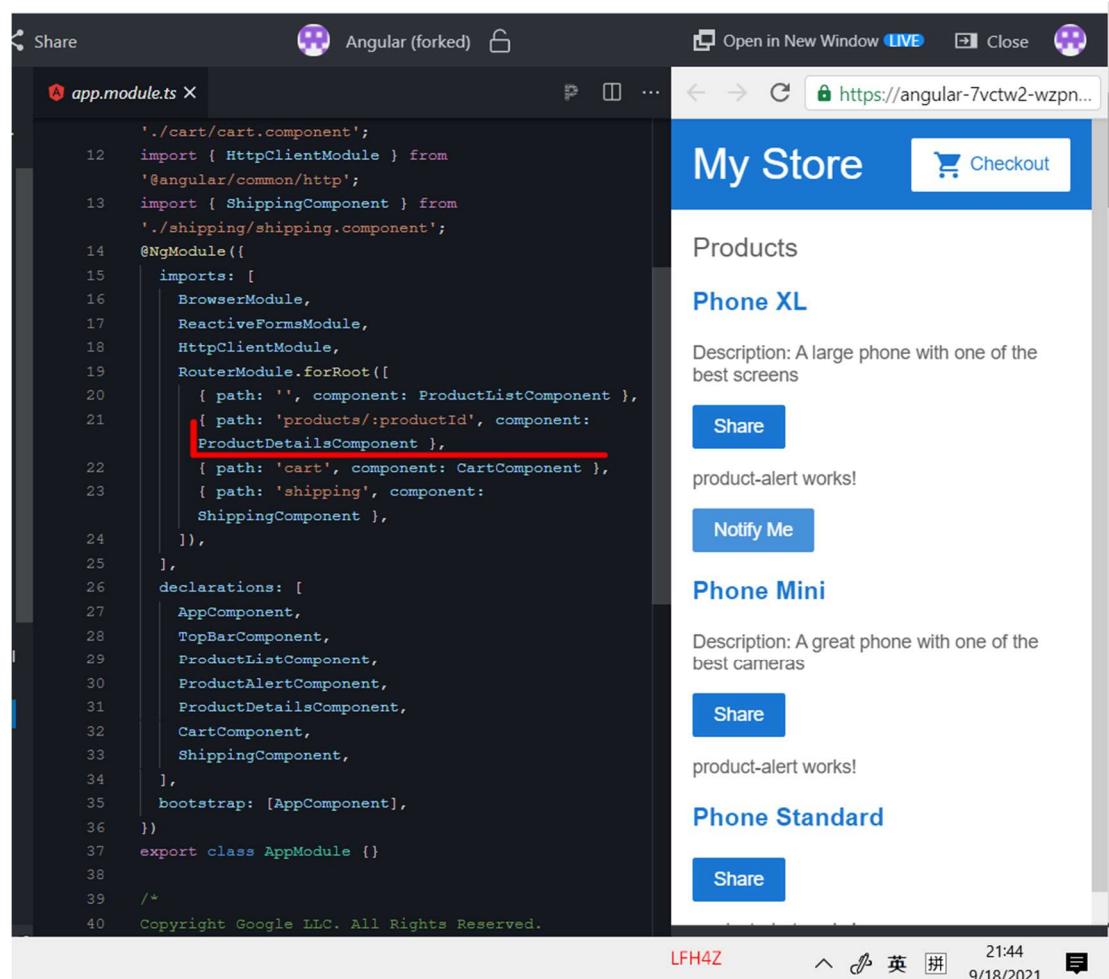
LFH4Z 21:42 9/18/2021

2 Navigate From One Page to Another

1) Create the product detail component to show the detail when clicking the items.



2) Add the component to the app.module.ts so that the component can be found and add the path to the app.module.ts file.



3) Add the link to the product-list page

The screenshot shows the Angular application running in a browser. The URL is <https://angular-7vctw2-wzpn...>. The page title is 'My Store'. It lists three products: 'Phone XL', 'Phone Mini', and 'Phone Standard'. Each product has a 'Share' button and a 'Notify Me' button. The 'Phone XL' product has a detailed description and a 'Description' button.

```

product-list.component.html
1 <h2>Products</h2>
2
3   <div *ngFor="let product of products">
4     <h3>
5       <a
6         [title]="product.name + ' details'"
7         [routerLink]=["'/products', product.id]"
8       >
9         {{ product.name }}
10        </a>
11      </h3>
12      <p *ngIf="product.description">Description: {{ product.description }}</p>
13      <button (click)="share()">Share</button>
14      <app-product-alert [product]="product" (notify)="onNotify()">
15        </app-product-alert>
16    </div>
17    <!--
18    Copyright Google LLC. All Rights Reserved.
19    Use of this source code is governed by an MIT-style
20    license that
21    can be found in the LICENSE file at
22    https://angular.io/license
23    -->

```

4) Import the Product class and product object to the product detail page

The screenshot shows the 'Product Details' page for the 'Phone XL'. The URL is <https://angular-7vctw2-wzpn...>. The page title is 'My Store'. It shows the product details for 'Phone XL' with a price of '\$799.00' and a 'Buy' button.

```

product-details.component.ts
1 import { Component, OnInit } from '@angular/core';
2 import { Product, products } from '../products';
3 import { ActivatedRoute } from '@angular/router';
4 import { CartService } from '../cart.service';
5
6 @Component({
7   selector: 'app-product-details',
8   templateUrl: './product-details.component.html',
9   styleUrls: ['./product-details.component.css'],
10 })
11 export class ProductDetailsComponent implements OnInit {
12   product: Product | undefined;
13   constructor(
14     private route: ActivatedRoute,
15     private cartService: CartService
16   ) {}
17   ngOnInit() {
18     // First get the product id from the current
19     // route.
20     const routeParams = this.route.snapshot.paramMap;
21     const productIdFromRoute = Number(routeParams.get('productId'));
22
23     // Find the product that correspond with the id
24     // provided in route.
25     this.product = products.find(function (product) {
26       return product.id === productIdFromRoute;
27     });
28
29     addToCart(product: Product) {
30       this.cartService.addToCart(product);
31       window.alert('Your product has been added to the
32       cart!');
33     }
34   }
35 }

```

5) Import ActivatedRoute so that the link information will be gotten

The screenshot shows a browser window with the URL <https://angular-7vctw2-wzpn...>. The page title is "My Store". On the right, there is a "Product Details" section for a "Phone XL" with a price of "\$799.00" and a description: "A large phone with one of the best screens". Below the description is a blue "Buy" button. On the left, the browser's developer tools are open, displaying the code for "product-details.component.ts". The code imports various Angular modules and services, including "ActivatedRoute" and "CartService". It defines a component selector "app-product-details" and an "OnInit" lifecycle hook. In the "ngOnInit" method, it retrieves the product ID from the route parameters and finds the corresponding product in the "products" array. Finally, it adds the product to the cart and shows a success message.

6) Use the `this.route.snapshot.paramMap` to get the link information and use `routeParams.get{'productId'}` to get the product id. Then use the gotten product id to get the product from the product.ts

The screenshot shows a browser window with the URL <https://angular-7vctw2-wzpn...>. The page title is "My Store". On the right, there is a "Product Details" section for a "Phone XL" with a price of "\$799.00" and a description: "A large phone with one of the best screens". Below the description is a blue "Buy" button. On the left, the browser's developer tools are open, displaying the code for "product-details.component.ts". The code imports various Angular modules and services, including "ActivatedRoute" and "CartService". It defines a component selector "app-product-details" and an "OnInit" lifecycle hook. In the "ngOnInit" method, it retrieves the product ID from the route parameters and finds the corresponding product in the "products" array. Finally, it adds the product to the cart and shows a success message. A red arrow points from the text "Get product from product.ts file" to the line of code where the product ID is retrieved from the route parameters.

7) Set the html content in the product detail page and create a button "buy" which call on the

function of the service to add this item to cart.

The screenshot shows a browser window with the Angular (forked) developer tools open. On the left, the code editor displays `product-details.component.html` with the following content:

```
1 <h2>Product Details</h2>
2
3 <div *ngIf="product">
4   <h3>{{ product.name }}</h3>
5   <h4>{{ product.price | currency }}</h4>
6   <p>{{ product.description }}</p>
7   <button (click)="addToCart(product)">Buy</button>
8 </div>
```

A red box highlights the entire `<div>` block, and a red arrow points from the text "Show the information gotten from product:" to the highlighted area. On the right, the browser shows the rendered "My Store" page for a "Phone XL". The product details are displayed: "Phone XL", "\$799.00", and "A large phone with one of the best screens". A blue "Buy" button is present.

8)add the routerLink to the product.name so that when clicking the product.name, the link will be redirected to the detail page.

The screenshot shows a browser window with the Angular (forked) developer tools open. On the left, the code editor displays `product-list.component.html` with the following content:

```
1 <h2>Products</h2>
2
3 <div *ngFor="let product of products">
4   <h3>
5     <a
6       [title]="product.name + ' details'"
7       [routerLink]=["/products", product.id]"
8     >
9       {{ product.name }}
10      </a>
11    </h3>
12    <p *ngIf="product.description">Description: {{ product.description }}</p>
13    <button (click)="share()>Share</button>
14    <app-product-alert [product]="product" (notify)="onNotify()>
15    </app-product-alert>
16  </div>
17  <!--
18  Copyright Google LLC. All Rights Reserved.
19  Use of this source code is governed by an MIT-style
20  license that
21  can be found in the LICENSE file at
22  https://angular.io/license
-->
```

On the right, the browser shows the "My Store" page. It lists a single product: "Phone XL" with a price of "\$799.00" and the description "A large phone with one of the best screens". A blue "Buy" button is present.

3. Using Service to Save Data

- 1) Create the cart service, import the object from product.ts, and create the function to be used within the export class. Then, import Cart Service to the product-detail component

The screenshot shows the Angular workspace in VS Code. A context menu is open over the `cart.service.ts` file, with the "Service" option highlighted. The code editor shows the `cart.service.ts` file containing the following code:

```
1 import { Injectable } from '@angular/core';
2 import { Product } from './products';
3 import { HttpClient } from '@angular/common/http';
4 @Injectable({
5   providedIn: 'root',
6 })
7 export class CartService {
8   items: Product[] = [];
9   constructor(private http: HttpClient) {}
10  addToCart(product: Product) {
11    this.items.push(product);
12  }
13  getItems() {
14    return this.items;
15  }
16  clearCart() {
17    this.items = [];
18    return this.items;
19  }
20  getShippingPrices() {
21    return this.http.get<{ type: string; price: number }[]>(
22      '/assets/shipping.json'
23    );
24  }
25 }
```

The browser preview on the right shows a product detail page for a "Phone XL" at \$799.00.

The screenshot shows the `cart.service.ts` file with the "Service" option removed from the context menu. The code has been modified to include the `clearCart` and `getShippingPrices` methods. The browser preview remains the same.

```
1 import { Injectable } from '@angular/core';
2 import { Product } from './products';
3 import { HttpClient } from '@angular/common/http';
4 @Injectable({
5   providedIn: 'root',
6 })
7 export class CartService {
8   items: Product[] = [];
9   constructor(private http: HttpClient) {}
10  addToCart(product: Product) {
11    this.items.push(product);
12  }
13  getItems() {
14    return this.items;
15  }
16  clearCart() {
17    this.items = [];
18    return this.items;
19  }
20  getShippingPrices() {
21    return this.http.get<{ type: string; price: number }[]>(
22      '/assets/shipping.json'
23    );
24  }
25 }
```

The screenshot shows a development environment with a code editor and a browser window.

Code Editor (cart.service.ts):

```
1 import { Injectable } from '@angular/core';
2 import { Product } from './products';
3 import { HttpClient } from '@angular/common/http';
4 @Injectable({
5   providedIn: 'root',
6 })
7 export class CartService {
8   items: Product[] = [];
9   constructor(private http: HttpClient) {}
10  addToCart(product: Product) {
11    this.items.push(product);
12  }
13  getItems() []
14  {
15    return this.items;
16  }
17  clearCart() {
18    this.items = [];
19    return this.items;
20  }
21  getShippingPrices() {
22    return this.http.get<{ type: string; price:
23      number }[]>(
24      '/assets/shipping.json'
25    );
26 }
```

Browser:

The browser displays a storefront interface titled "My Store". A product detail page for "Phone XL" is shown, featuring a large image, the title "Phone XL", the price "\$799.00", a description "A large phone with one of the best screens", and a blue "Buy" button.

Console

LFH4Z

22:04
9/18/2021

The screenshot shows a code editor with a file named `product-details.component.ts` open. The code is an Angular component that imports `@Component`, `OnInit`, `Product`, `products`, `ActivatedRoute`, and `CartService`. It defines a selector for the component and implements `OnInit`. In the constructor, it injects `ActivatedRoute` and `CartService`. The `ngOnInit` method retrieves the product ID from the route parameters, finds the corresponding product in the `products` array, and adds it to the cart using the `CartService`. A success message is displayed via an alert.

```
1 import { Component, OnInit } from '@angular/core';
2 import { Product, products } from '../products';
3 import { ActivatedRoute } from '@angular/router';
4 import { CartService } from '../cart.service';
5 @Component({
6   selector: 'app-product-details',
7   templateUrl: './product-details.component.html',
8   styleUrls: ['./product-details.component.css'],
9 })
10 export class ProductDetailsComponent implements
11   OnInit {
12   product: Product | undefined;
13   constructor(
14     private route: ActivatedRoute,
15     private cartService: CartService
16   ) {}
17   ngOnInit() {
18     // First get the product id from the current
19     // route.
20     const routeParams = this.route.snapshot.paramMap;
21     const productIdFromRoute = Number(routeParams.get(
22       'productId'));
23
24     // Find the product that correspond with the id
25     // provided in route.
26     this.product = products.find(function (product) {
27       return product.id === productIdFromRoute;
28     });
29   }
30
31   addToCart(product: Product) {
32     this.cartService.addToCart(product);
33     window.alert('Your product has been added to the
34     cart!');
35   }
36 }
```

The right side of the interface shows a browser preview of the application. The title bar says "My Sto". The main content displays a product card for "Phone XL". The card includes the product name, price (\$799.00), a description ("A large phone!"), and a blue "Buy" button.

2)Add function in the product detail component to use the service

The screenshot shows a browser window with the URL 'Angular (forked)'. The main content area displays a product detail page for a 'Phone XL'. The page includes a title 'My Store', a heading 'Product Details', the product name 'Phone XL', its price '\$799.00', a description 'A large phone', and a blue 'Buy' button. On the left side of the browser, there is a code editor showing the 'product-details.component.ts' file. The code is written in TypeScript and defines a component that implements the OnInit interface. It retrieves a product ID from the route parameters, finds the corresponding product in a list, and adds it to the cart using a service. A red box highlights the 'addToCart' method and its alert message.

```
6   selector: 'app-product-details',
7   templateUrl: './product-details.component.html',
8   styleUrls: ['./product-details.component.css'],
9 }
10 export class ProductDetailsComponent implements
11   OnInit {
12   product: Product | undefined;
13   constructor(
14     private route: ActivatedRoute,
15     private cartService: CartService
16   ) {}
17   ngOnInit() {
18     // First get the product id from the current
19     // route.
20     const routeParams = this.route.snapshot.paramMap;
21     const productIdFromRoute = Number(routeParams.get(
22       'productId'));
23
24     // Find the product that correspond with the id
25     // provided in route.
26     this.product = products.find(function (product) {
27       return product.id === productIdFromRoute;
28     });
29   }
30   addToCart(product: Product) {
31     this.cartService.addToCart(product);
32     window.alert('Your product has been added to the
33     cart!');
34   }
35 }
```

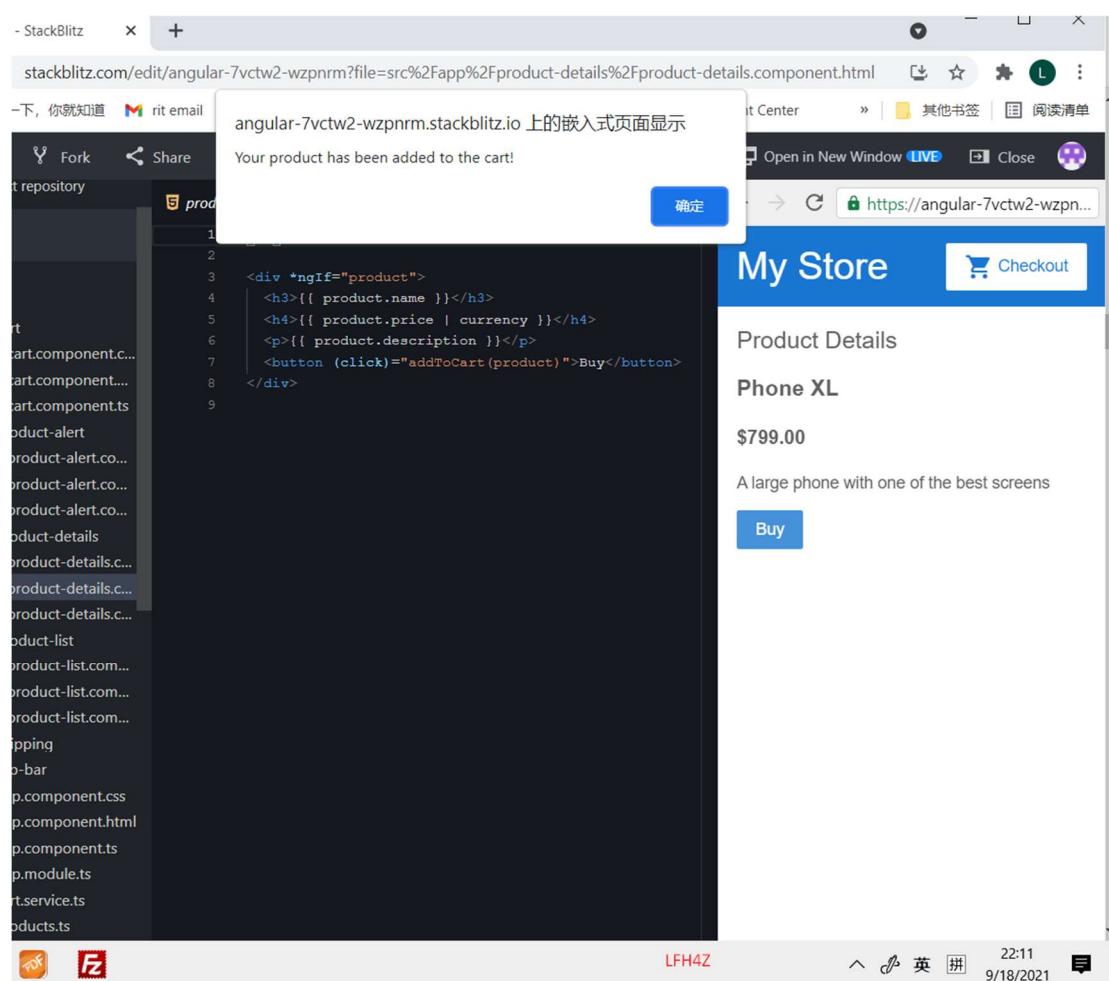
3)Create the button on the product detail page to use the function of the product detail component

The screenshot shows a browser window with the Angular (forked) tab open. On the left, the product-details.component.html file is displayed with the following code:

```
1 <h2>Product Details</h2>
2
3 <div *ngIf="product">
4   <h3>{{ product.name }}</h3>
5   <h4>{{ product.price | currency }}</h4>
6   <p>{{ product.description }}</p>
7   <button (click)="addToCart(product)">Buy</button>
8 </div>
```

On the right, the browser displays the "My Store" application. It shows a product detail page for a "Phone XL" with a price of \$799.00. A "Buy" button is present. The URL in the address bar is <https://angular-7vctw2-wzpn...>.

4) Verify the button



5) create the cart component to show the view of the cart

```
cart.component.html
1 <h3>Cart</h3>
2 <p>
3 | <a routerLink="/shipping">Shipping Prices</a>
4 </p>
5 <div class="cart-item" *ngFor="let item of items">
6 | <span>{{ item.name }}</span>
7 | <span>{{ item.price | currency }}</span>
8 </div>
9 <form [formGroup]="checkoutForm" (ngSubmit)="onSubmit()">
10 <div>
11 | <label for="name"> Name </label>
12 | <input id="name" type="text"
13 | formControlName="name" />
14 </div>
15 <div>
16 | <label for="address"> Address </label>
17 | <input id="address" type="text"
18 | formControlName="address" />
19 </div>
20 <button class="button"
21 type="submit">Purchase</button>
22 </form>
```

6) set the html content in the cart component

My Store

Product Details

Phone XL

\$799.00

A large phone with one of the best screens

Buy

7) Set the URL for the cart page.

```

    ...
    9 import { ProductAlertComponent } from
    10 import { ProductDetailsComponent } from
    11 import { CartComponent } from
    12 import { HttpClientModule } from
    13 import { ShippingComponent } from
    14 import { NgModule } from
    15 imports: [
    16   BrowserModule,
    17   ReactiveFormsModule,
    18   HttpClientModule,
    19   RouterModule.forRoot([
    20     { path: '', component: ProductListComponent },
    21     { path: 'products/:productId', component:
    22       ProductDetailsComponent },
    23     { path: 'cart', component: CartComponent },
    24     { path: 'shipping', component:
    25       shippingComponent },
    26   ]),
    27   declarations: [
    28     AppComponent,
    29     TopBarComponent,
    30     ProductListComponent,
    31     ProductAlertComponent,
    32     ProductDetailsComponent,
    33     CartComponent,
    34     ShippingComponent,
    35   ],
  
```

8) Verify the progress by clicking the checkout page which contains the items added

```

    ...
    1 <h3>Cart</h3>
    2 <p>
    3   <a href="/shipping">Shipping Prices</a>
    4 </p>
    5 <div class="cart-item" *ngFor="let item of items">
    6   <span>{{ item.name }}</span>
    7   <span>{{ item.price | currency }}</span>
    8 </div>
    9 <form [formGroup]="checkoutForm" (ngSubmit)="onSubmit()">
    10   <div>
    11     <label for="name">Name </label>
    12     <input id="name" type="text" formControlName="name" />
    13   </div>
    14 <div>
    15   ...
  
```

4. Using Http to Get Data

1) Create shipping component and create the shipping.json file

The screenshot shows the VS Code interface with the file `shipping.component.ts` open. The code defines a component named `ShippingComponent` that implements `OnInit`. It imports `Component` from `@angular/core` and `CartService` from `../cart.service`. The component has a selector of `'app-shipping'`, a template URL of `'./shipping.component.html'`, and style URLs of `'./shipping.component.css'`. In the constructor, it takes a `CartService` dependency and initializes `shippingCosts` with the result of calling `getShippingPrices`. The `ngOnInit` method is also defined.

```
1 import { Component, OnInit } from '@angular/core';
2 import { CartService } from '../cart.service';
3 @Component({
4   selector: 'app-shipping',
5   templateUrl: './shipping.component.html',
6   styleUrls: ['./shipping.component.css'],
7 })
8 export class ShippingComponent implements OnInit {
9   constructor(private cartService: CartService) {}
10   shippingCosts = this.cartService.getShippingPrices();
11   ngOnInit() {}
12 }
```

The screenshot shows the VS Code interface with the file `shipping.json` open. The JSON array contains three objects representing shipping types: "Overnight" (price 25.99), "2-Day" (price 9.99), and "Postal" (price 2.99). A red box highlights the first object in the array.

```
[{"type": "Overnight", "price": 25.99}, {"type": "2-Day", "price": 9.99}, {"type": "Postal", "price": 2.99}]
```

2) Import `CartService` and create an object in the `constructor()`. Then use this service object to get shipping price.



```
shipping.component.ts X
1 import { Component, OnInit } from '@angular/core';
2 import { CartService } from '../cart.service';
3 @Component({
4   selector: 'app-shipping',
5   templateUrl: './shipping.component.html',
6   styleUrls: ['./shipping.component.css'],
7 })
8 export class ShippingComponent implements OnInit {
9   constructor(private cartService: CartService) {}
10  shippingCosts = this.cartService.getShippingPrices();
11  ngOnInit() {}
12 }
13
```

3) Set up the html file to show the content of shipping on the shipping page



```
shipping.component.html X
1 <h3>Shipping Prices</h3>
2
3 <div class="shipping-item" *ngFor="let shipping of shippingCosts | async">
4   <span>{{ shipping.type }}</span>
5   <span>{{ shipping.price | currency }}</span>
6 </div>
```

4) Set up a link on the cart page which links the shipping page

The screenshot shows a browser window with a dark theme. On the left, a sidebar lists Angular components: app, cart, cart.component.css, cart.component.ts, cart.component.html, product-alert, product-alert.co..., product-alert.co..., product-alert.co..., product-details, product-details.c..., product-details.c..., product-details.c..., product-list, product-list.com..., product-list.com..., product-list.com..., shipping, shipping.compon..., shipping.compon..., shipping.compon..., top-bar, and app.component.css. Below this is a message: "Unlock Upgrades! Go 🎉". The main content area displays a form titled "Cart" with fields for "NAME" and "ADDRESS", and a "Purchase" button. The URL in the address bar is "http://localhost:4200/cart".

```

1 <h3>Cart</h3>
2 <p>
3 | <a routerLink="/shipping">Shipping Prices</a>
4 </p>
5 <div class="cart-item" *ngFor="let item of items">
6 <span>{{ item.name }}</span>
7 <span>{{ item.price | currency }}</span>
8 </div>
9 <form [FormGroup]="checkoutForm" (ngSubmit)="onSubmit()">
10 <div>
11 | <label for="name"> Name </label>
12 | <input id="name" type="text" formControlName="name" />
13 </div>
14 <div>
15 | <label for="address"> Address </label>
16 | <input id="address" type="text" formControlName="address" />
17 </div>
18 <button class="button" type="submit">Purchase</button>
19 </form>
20
21
22

```

5) Import the HttpClient and create an object in the constructor()

The screenshot shows a browser window with a light theme. On the left, a sidebar lists Angular components: app, cart, cart.service.ts, cart.component.css, cart.component.ts, cart.component.html, product-alert, product-alert.co..., product-alert.co..., product-alert.co..., product-details, product-details.c..., product-details.c..., product-details.c..., product-list, product-list.com..., product-list.com..., product-list.com..., shipping, shipping.compon..., shipping.compon..., shipping.compon..., top-bar, and app.component.css. Below this is a message: "Unlock Upgrades! Go 🎉". The main content area displays a form titled "Cart" with fields for "NAME" and "ADDRESS", and a "Purchase" button. The URL in the address bar is "http://localhost:4200/cart".

```

1 import { Injectable } from '@angular/core';
2 import { Product } from './products';
3 import { HttpClient } from '@angular/common/http';
4 @Injectable({
5   providedIn: 'root',
6 })
7 export class CartService {
8   items: Product[] = [];
9   constructor(private http: HttpClient) {}
10 addToCart(product: Product) {
11   this.items.push(product);
12 }
13 getItems() {
14   return this.items;
15 }
16 clearCart() {
17   this.items = [];
18   return this.items;
19 }
20 getShippingPrices() {
21   return this.http.get<{ type: string; price: number }[]>(
22     '/assets/shipping.json'
23   );
24 }
25
26

```

6) Add the HttpClientModule to the modul.ts file

StackBlitz

ackblitz.com/edit/angular-7vctw2-wzpnrm?file=src%2Fapp%2Fapp.module.ts

你就知道 M rit email Y 有道首页 Employee-facing re... Billing Center Login Student Center » | 其他书签 |

Fork Share Angular (forked) Open in New Window LIVE Close

app.module.ts

```

5
6 import { AppComponent } from './app.component';
7 import { TopBarComponent } from './top-bar/top-bar.component';
8 import { ProductListComponent } from './product-list/product-list.component';
9 import { ProductAlertComponent } from './product-alert/product-alert.component';
10 import { ProductDetailsComponent } from './product-details/product-details.component';
11 import { CartComponent } from './cart/cart.component';
12 import { HttpClientModule } from '@angular/common/http';
13 import { ShippingComponent } from './shipping/shipping.component';
14 @NgModule({
15   imports: [
16     BrowserModule,
17     ReactiveFormsModule,
18     HttpClientModule,
19     RouterModule.forRoot([
20       { path: '', component: ProductListComponent },
21       { path: 'products/:productId', component: ProductDetailsComponent },
22       { path: 'cart', component: CartComponent },
23       { path: 'shipping', component: ShippingComponent },
24     ]),
25   ],
26   declarations: [
27     AppComponent,
28     TopBarComponent,
29     ProductListComponent,
30     ProductAlertComponent,
31     ProductDetailsComponent,
32     CartComponent,
33     ShippingComponent,
34   ],
35   bootstrap: [AppComponent],

```

Upgrades! Go

LFH4Z 22:25 9/18/2021

My Store

Cart

Shipping Prices

Phone XL

Phone XL

NAME

ADDRESS

Purchase

Console

7) Verify the shipping page by clicking the link

Share Angular (forked) Open in New Window LIVE Close

cart.component.html

[Cart](#)

[Shipping Prices](#)

Cart

Shipping Prices

Phone XL \$799.00

Phone XL \$799.00

NAME

ADDRESS

Purchase

Checkout

LFH4Z 22:34 9/18/2021

cart.component.html

```

1 <h3>Cart</h3>
2 <p>
3   <a routerLink="/shipping">Shipping Prices</a>
4 </p>
5 <div class="cart-item"
6   *ngFor="let item of items">
7     <span>{{ item.name }}</span>
8     <span>{{ item.price | currency }}</span>
9   </div>
10 <form [formGroup]
11   ="checkoutForm" (ngSubmit)
12   ="onSubmit()">
13   <div>
14     <label for="name"> Name
15     </label>
16     <input id="name"
17       type="text"
18       formControlName="name" />
19   </div>
20
21   <div>
22     <label for="address">
23       Address </label>
24     <input id="address"
25       type="text"
26       formControlName="address" />
27   </div>
28
29
30
31
32
33
34
35
36
37
38
39

```

Shipping Prices

Overnight	\$25.99
2-Day	\$9.99
Postal	\$2.99

Console

LFH4Z 22:35 9/18/2021

5. Using form

1) Import the "FormBuilder" component and create a object in the constructor()

cart.component.ts

```

1 import { Component, OnInit } from '@angular/core';
2 import { CartService } from '../cart.service';
3 import { FormBuilder } from '@angular/forms';
4
5 @Component({
6   selector: 'app-cart',
7   templateUrl: './cart.component.html',
8   styleUrls: ['./cart.component.css'],
9 })
10 export class CartComponent implements OnInit {
11   items = this.cartService.getItems();
12   checkoutForm = this.formBuilder.group({
13     name: '',
14     address: '',
15   });
16   constructor(
17     private cartService: CartService,
18     private formBuilder: FormBuilder
19   ) {}
20   onSubmit(): void {
21     // Process checkout data here
22     this.items = this.cartService.clearCart();
23     console.warn('Your order has been submitted');
24     this.checkoutForm.value;
25     this.checkoutForm.reset();
26   }
27   ngOnInit() {}
28 }

```

My Store

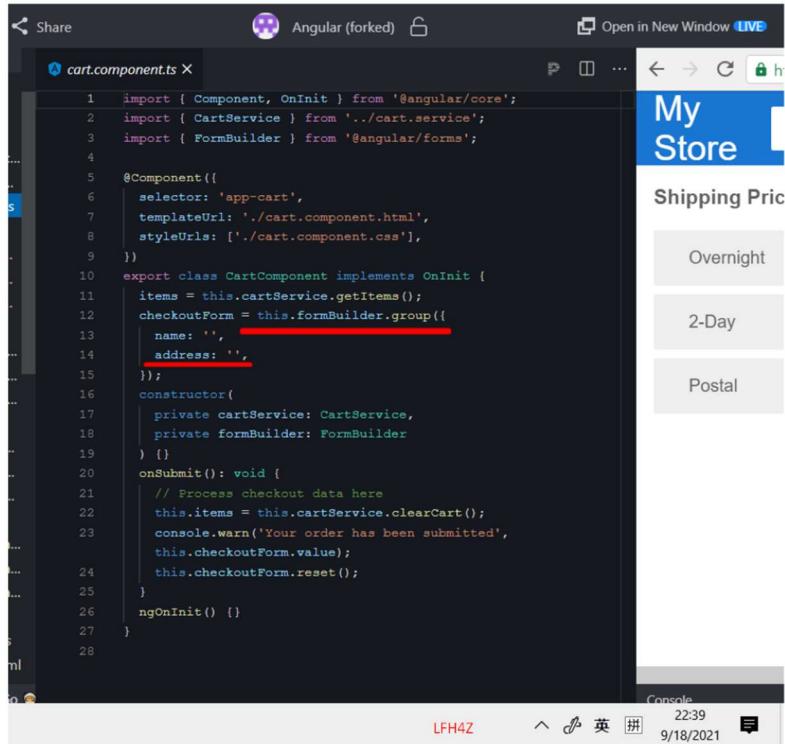
Shipping Prices

Overnight	
2-Day	
Postal	

Console

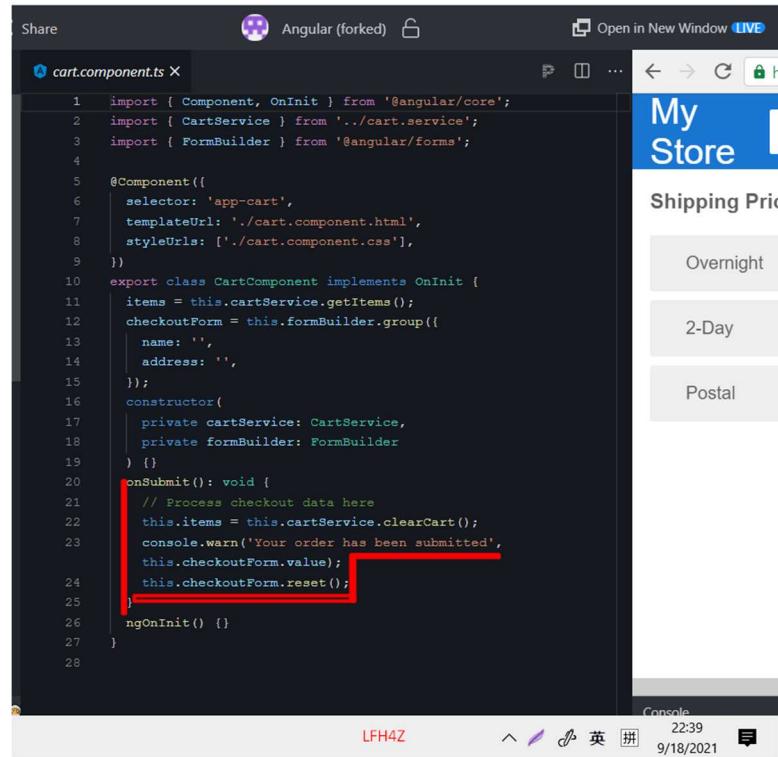
LFH4Z 22:36 9/18/2021

2)Create an object to set up the properties of the form.



```
1 import { Component, OnInit } from '@angular/core';
2 import { CartService } from '../cart.service';
3 import { FormBuilder } from '@angular/forms';
4
5 @Component({
6   selector: 'app-cart',
7   templateUrl: './cart.component.html',
8   styleUrls: ['./cart.component.css'],
9 })
10 export class CartComponent implements OnInit {
11   items = this.cartService.getItems();
12   checkoutForm = this.formBuilder.group({
13     name: '',
14     address: '',
15   });
16   constructor(
17     private cartService: CartService,
18     private formBuilder: FormBuilder
19   ) {}
20   onSubmit(): void {
21     // Process checkout data here
22     this.items = this.cartService.clearCart();
23     console.warn('Your order has been submitted',
24       this.checkoutForm.value);
25     this.checkoutForm.reset();
26   }
27   ngOnInit() {}
28 }
```

3)Create onSubmit() function to flush the content of the cart when the form is submitted.



```
1 import { Component, OnInit } from '@angular/core';
2 import { CartService } from '../cart.service';
3 import { FormBuilder } from '@angular/forms';
4
5 @Component({
6   selector: 'app-cart',
7   templateUrl: './cart.component.html',
8   styleUrls: ['./cart.component.css'],
9 })
10 export class CartComponent implements OnInit {
11   items = this.cartService.getItems();
12   checkoutForm = this.formBuilder.group({
13     name: '',
14     address: '',
15   });
16   constructor(
17     private cartService: CartService,
18     private formBuilder: FormBuilder
19   ) {}
20   onSubmit(): void {
21     // Process checkout data here
22     this.items = this.cartService.clearCart();
23     console.warn('Your order has been submitted',
24       this.checkoutForm.value);
25     this.checkoutForm.reset();
26   }
27   ngOnInit() {}
28 }
```

4)Set up the html content in the cart component

The screenshot shows a browser window with the title "My Store". On the left, the code editor displays the component template (`cart.component.html`). The template includes a heading "Cart", a link to "Shipping Prices", and a form for entering shipping details. The form contains fields for "Name" and "Address", and a "Purchase" button. The "Purchase" button is highlighted with a red rectangle. On the right, the browser displays the "Cart" page with a "Shipping Prices" section. This section lists three options: "Overnight", "2-Day", and "Postal". The "2-Day" option is currently selected and highlighted with a gray background.

```

1 <h3>Cart</h3>
2 <p>
3 | <a routerLink="/shipping">Shipping Prices</a>
4 </p>
5 <div class="cart-item" *ngFor="let item of items">
6 | <span>{{ item.name }}</span>
7 | <span>{{ item.price | currency }}</span>
8 </div>
9 <form [formGroup]="checkoutForm" (ngSubmit)="onSubmit()">
10 <div>
11 | <label for="name"> Name </label>
12 | <input id="name" type="text" formControlName="name" />
13 </div>
14 <div>
15 | <label for="address"> Address </label>
16 | <input id="address" type="text" formControlName="address" />
17 </div>
18 <button class="button" type="submit">Purchase</button>
19 </form>
20
21
22

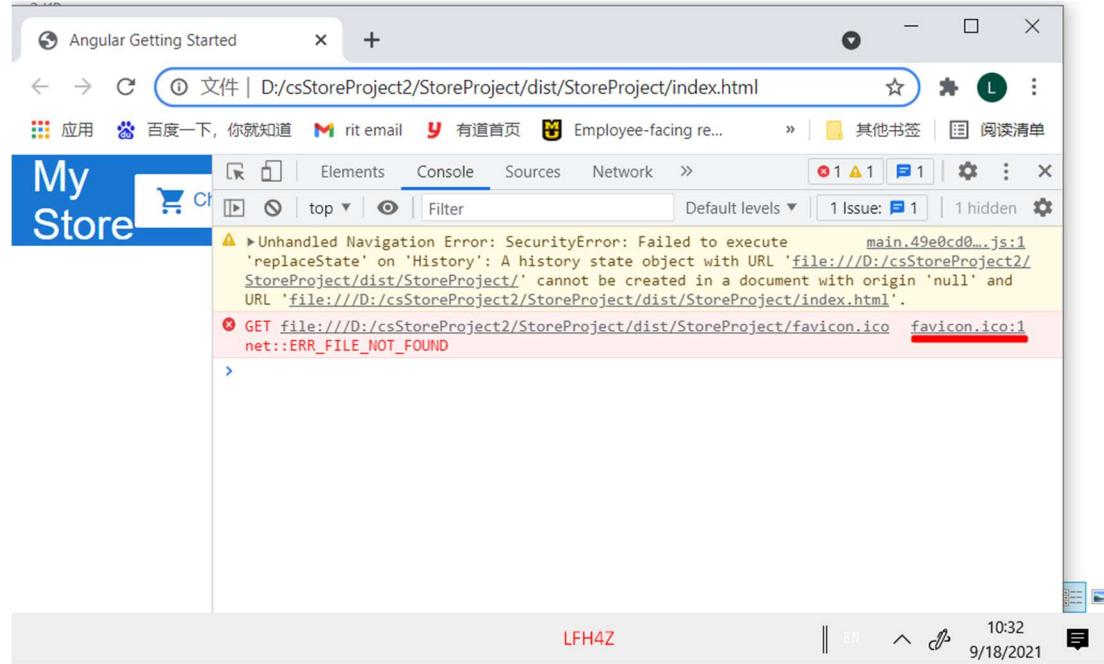
```

5) Verify the form

The screenshot shows a browser window with the URL `https://angular-7vctw2-wzpnrm.stackblitz.io/cart`. The page displays the "Cart" section and the "Shipping Prices" section. In the "Shipping Prices" section, the "2-Day" option is selected. Below it, a form is displayed for entering shipping details. The form has fields for "NAME" and "ADDRESS", both of which are populated with placeholder text ("lihaofan" and "missouri65201" respectively). A "Purchase" button is at the bottom of the form. The "Purchase" button is highlighted with a red rectangle. The browser status bar at the bottom shows "LFH4Z" and the date "9/18/2021".

6. Deploy Error

- I tried to use `ng build` to transform the project to static website. But I encountered the following `favicon.ico` error. I fixed it by adding an image, named "favicon.ico".



2) When I tried to run the static website on my web server. It occurs the following error. This is because <base href="/"> can't guide the webpage to correct resources. I fixed it by deleting it and adding <base href="http://www.lihaofan.me/exploration1/StoreProject/">.

