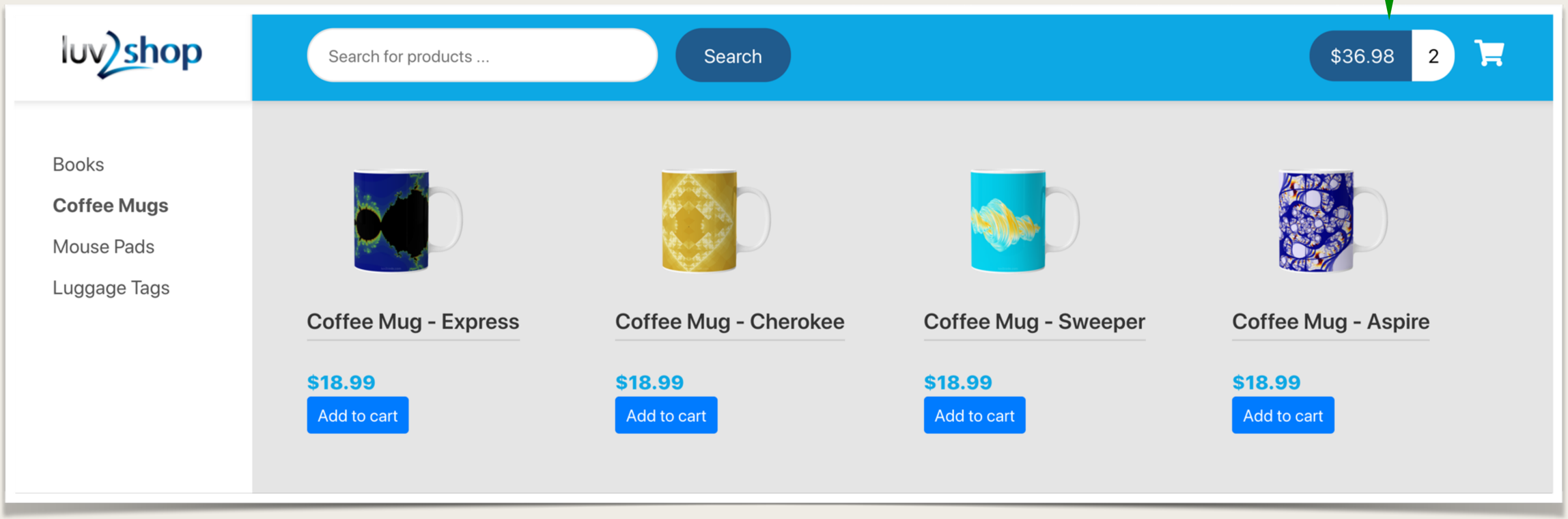
Add Products to Shopping Cart - Part 2

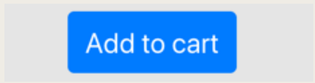
Add Products to Shopping Cart



Development Process - Part 2

1. Create model class: CartItem
2. Develop CartService
3. Modify ProductListComponent to call CartService
4. Enhance CartStatusComponent to subscribe to CartService
5. Update CartStatusComponent HTML to display cart total price and quantity

Application Interaction



ProductListComponent

CartStatusComponent

1. subscribe for events

2. addToCart(...)

3. publish events to all subscribers

4. update UI for total price and quantity

CartService

Step 1: Create model class: CartItem

File: cart-item.ts

```
export class CartItem {

    id: string;
    name: string;
    imageUrl: string;
    unitPrice: number;
    quantity: number;

    constructor(product: Product) {
        this.id = product.id;
        this.name = product.name;
        this.imageUrl = product.imageUrl;
        this.unitPrice = product.unitPrice;
        this.quantity = 1;
    }
}
```

Contains essential fields of Product for use in the cart

Also has a field for quantity

We have "x" number of "widgets" in our cart

Step 2: Develop CartService

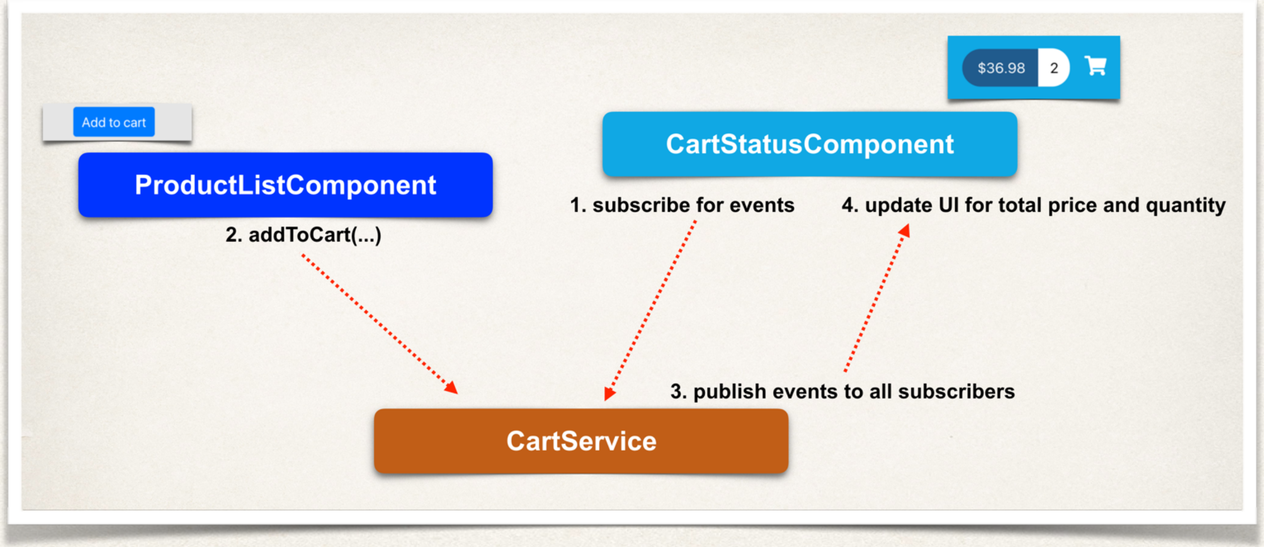
File: cart-service.ts

```
export class CartService {

  cartItems: CartItem[] = [];

  totalPrice: Subject<number> = new Subject<number>();
  totalQuantity: Subject<number> = new Subject<number>();

  ...
}
```

`Subject` is a subclass of `Observable`

We can use `Subject` to publish events in our code.

The event will be sent to all of the subscribers



Step 2: Develop CartService

File: cart-service.ts

```
export class CartService {

  cartItems: CartItem[] = [];
  ...

  addToCart(theCartItem: CartItem) {

    // check if we already have the item in our cart
    let alreadyExistsInCart: boolean = false;
    let existingCartItem: CartItem = undefined;

    if (this.cartItems.length > 0) {
      // find the item in the cart based on item id

      for (let tempCartItem of this.cartItems) {
        if (tempCartItem.id === theCartItem.id) {
          existingCartItem = tempCartItem;
          break;
        }
      }

      // check if we found it
      alreadyExistsInCart = (existingCartItem != undefined)
    }
```

```
    if (alreadyExistsInCart) {
      // increment the quantity
      existingCartItem.quantity++;
    }
    else {
      // just add the item to the array
      this.cartItems.push(theCartItem);
    }

    // compute cart quantity and cart total
    this.computeCartTotals();
  }

  ...
}
```

Step 2: Develop CartService

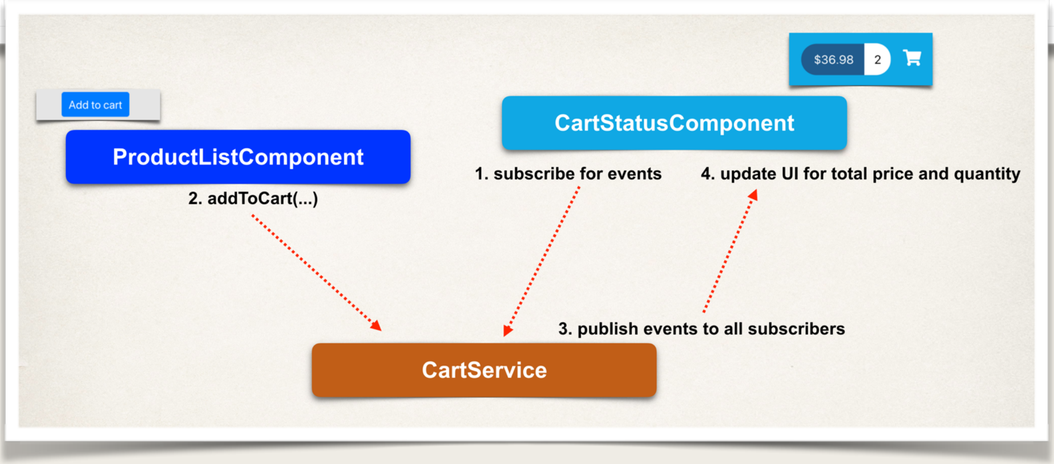
File: cart-service.ts

```
computeCartTotals() {

  let totalPriceValue: number = 0;
  let totalQuantityValue: number = 0;

  for (let currentCartItem of this.cartItems) {
    totalPriceValue += currentCartItem.quantity * currentCartItem.unitPrice;
    totalQuantityValue += currentCartItem.quantity;
  }

  // publish the new values ... all subscribers will receive the new data
  this.totalPrice.next(totalPriceValue);
  this.totalQuantity.next(totalQuantityValue);
}
```

Compute totals

This will publish events to all subscribers

one event for totalPrice
one event for totalQuantity

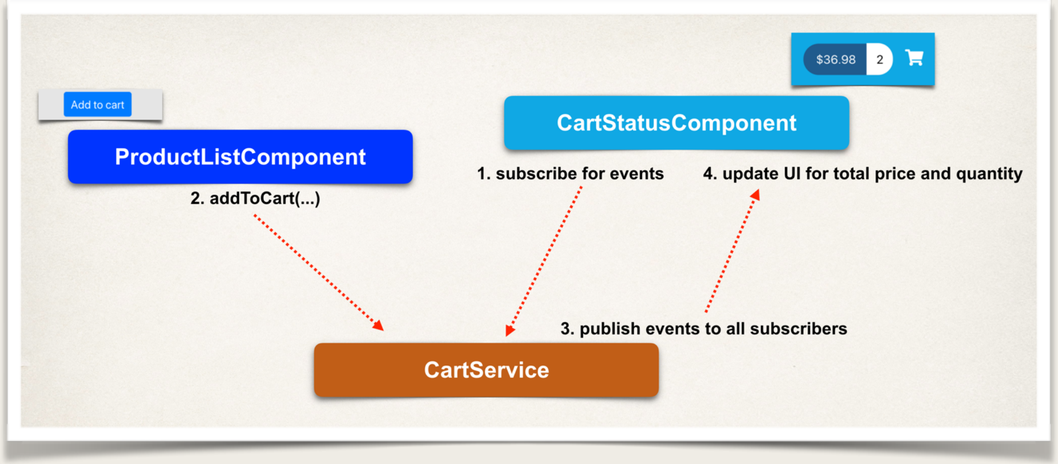


Step 3: Modify ProductListComponent to call CartService

File: product-list.component.ts

```
addToCart(theProduct: Product) {
  console.log(`Adding to cart: ${theProduct.name}, ${theProduct.unitPrice}`);

  const theCartItem = new CartItem(theProduct);

  // TODO ... do the real work
  this.cartService.addToCart(theCartItem);
}
```

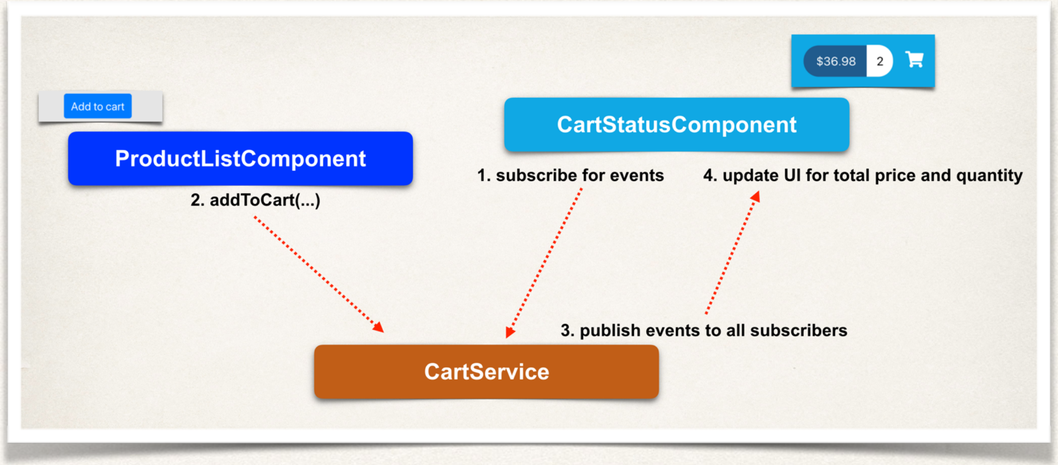
Call CartService



Step 4: Enhance CartStatusComponent to subscribe to CartService

File: cart-status.component.ts

```
export class CartStatusComponent implements OnInit {

  totalPrice: number = 0.00;
  totalQuantity: number = 0;

  constructor(private cartService: CartService) { }

  ngOnInit(): void {
    this.updateCartStatus();
  }

  updateCartStatus() {

    // subscribe to the cart status totalPrice
    this.cartService.totalPrice.subscribe(
      data => this.totalPrice = data
    );

    // subscribe to the cart status totalQuantity
    this.cartService.totalQuantity.subscribe(
      data => this.totalQuantity = data
    );
  }
}
```



When new events are received, make the assignments to update UI

Step 5: Update CartStatusComponent HTML to display cart total price and quantity

File: cart-status.component.html

```
<div class="total">{{ totalPrice | currency: 'USD' }}
        <span>{{ totalQuantity }}</span>
</div>
```