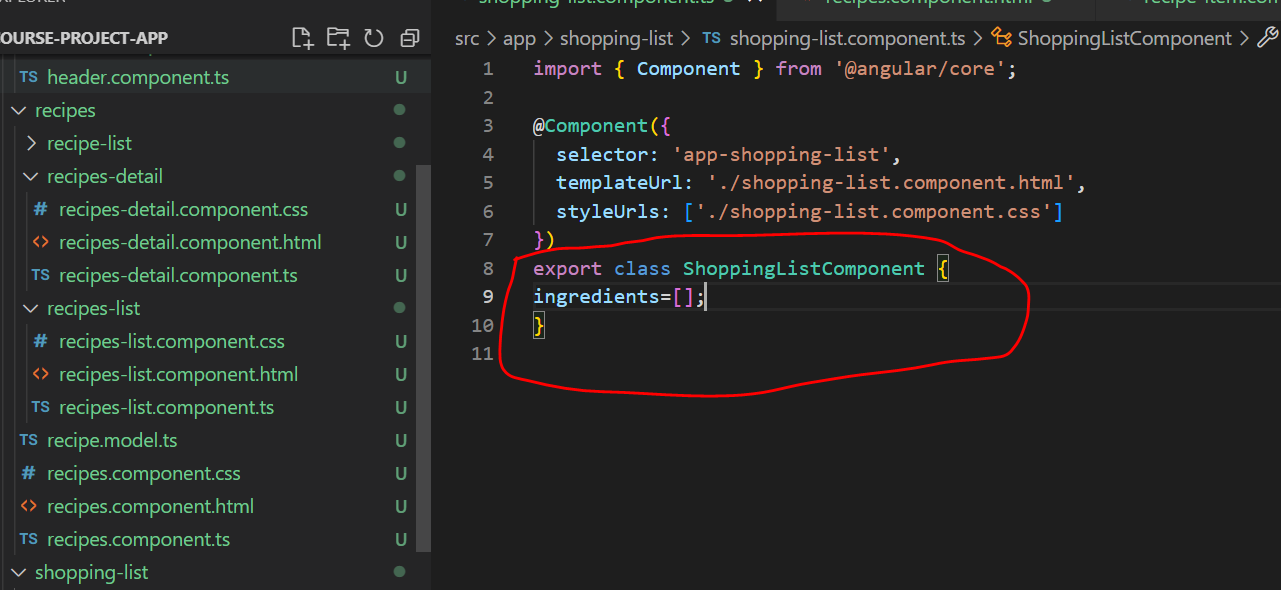
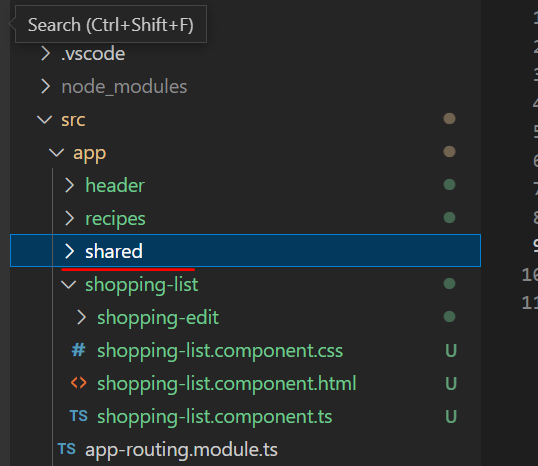
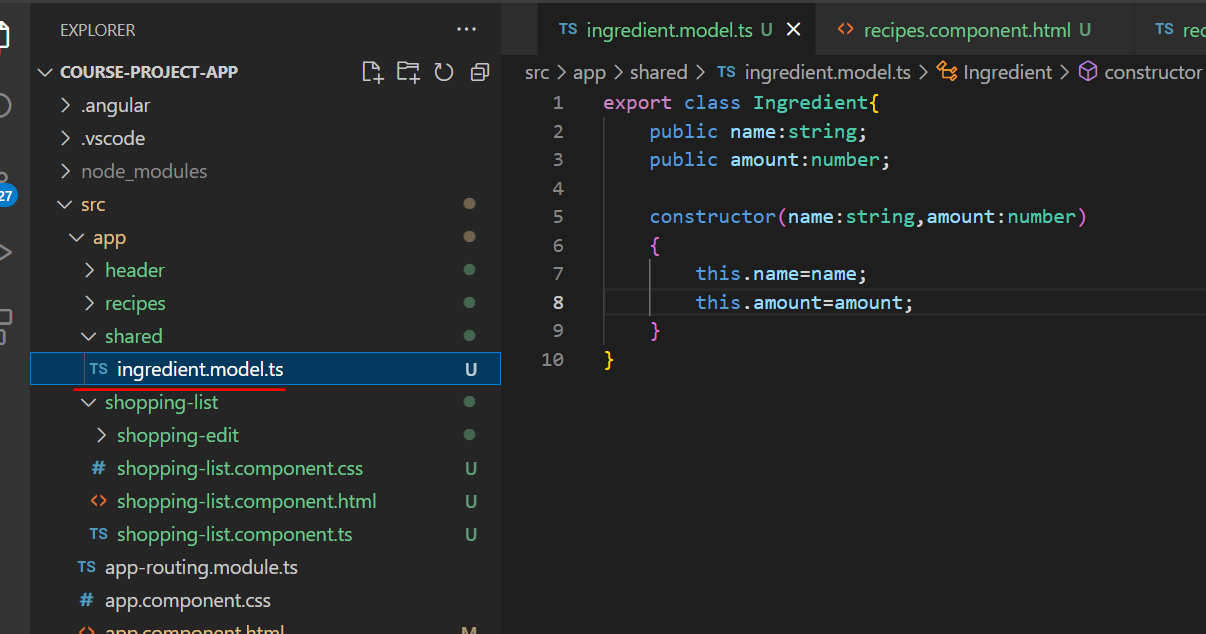
1. **Working on Shopping List Component:**

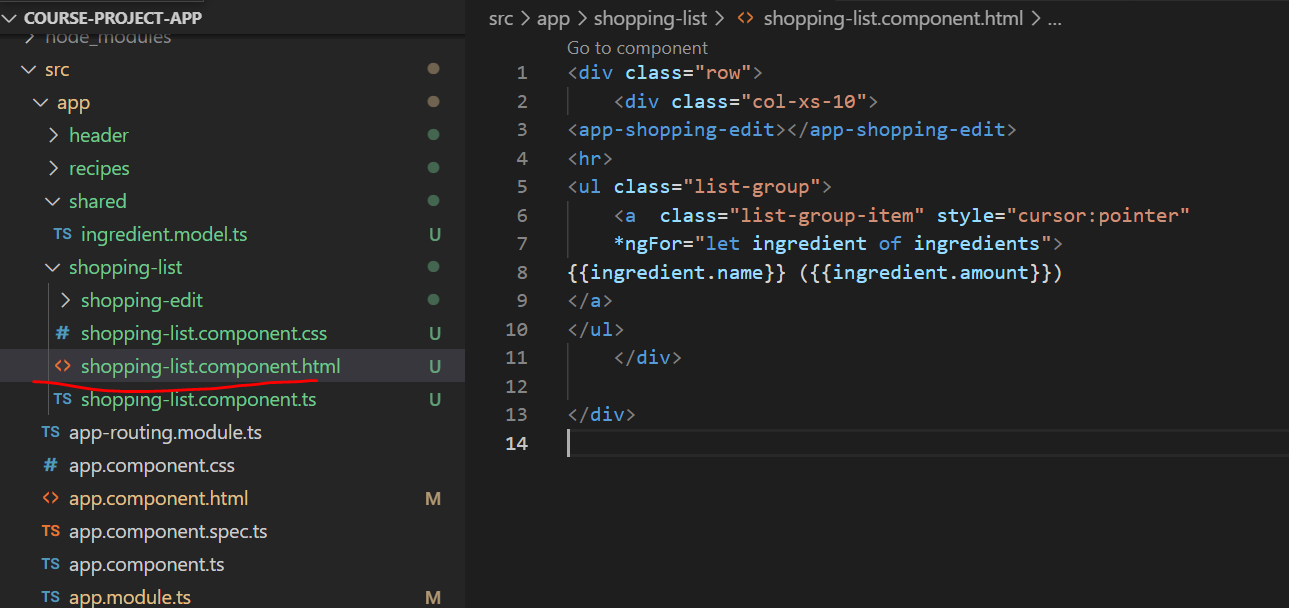
* Let's work on the shopping list now.
* In html file(shopping-list.component.html), Added the following changes
* <div class="row">
* <div class="col-xs-10">
* <app-shopping-edit></app-shopping-edit>
* <hr>
* <ul class="list-group">
* <a  class="list-group-item" style="cursor:pointer"></a>
* </ul>
* </div>
* </div>
* Now, to display something here we need an array of ingredients though.
* So what I'll do for now is I'll add my ingredients property here to the shopping list, and this will be an empty array.
* But now just like with the recipe we're going to use ingredients a lot throughout our app, also in the recipe section.
* So we should create a model for ingredient.



1. **Creating Ingredient model:**

* We store the recipe model in the recipes folder because it belongs there feature-wise.
* Similarly ingredients model will be stored in shared folder.
* 
* Because shared folder will contain features or elements of our app which are shared across different features.
* Ingredient model will be used both in the shopping list and the recipe section.
* In shared folder, ingredient.model.ts file will be created and export a class named ingredient and define the properties of ingredient.
* 
* **Shortcut method:**
* We can get rid of the declaration of our properties up here and get rid of the content in the body of this constructor and simply add an accessor in front of the property name here - public in front of both arguments.
* Behind the scenes it will automatically give us properties with the names we specify here as argument names. So name and amount in this case. And it'll automatically assign the values we receive in this constructor to these newly created properties.
* Eg:
* export class Ingredient{
* constructor(public name:string, public amount:number)
* {}
* }

1. **Creating and Outputting Shopping List:**

* Let's use our newly created ingredient model. So just like with recipe, we can now define the type for our ingredients array,
* import { Component } from '@angular/core';
* import { Ingredient } from '../shared/ingredient.model';
* @Component({
* selector: 'app-shopping-list',
* templateUrl: './shopping-list.component.html',
* styleUrls: ['./shopping-list.component.css']
* })
* export class ShoppingListComponent {
* ingredients:Ingredient[]=[
* new Ingredient('Apples',5),
* new Ingredient('Oranges',10)
* ];
* }
* Now we want to output it here in the component(shopping-list.component.html).
* 
* Now when we check the same in browser we get:

Graphical user interface, application

Description automatically generated

* The last missing piece I want to addis this shopping list edit section here,which should display us a input field,which then allows us to actually edit our itemsor add new ones

1. **Adding a shopping list edit section:**

* let's work on the shopping list edit component.
* In it's template(shopping-edit.component.html) lets make the following changes
* <div class="row">
* <div class="col-xs-12">
* <form>
* <div class="col-sm-5 for-group">
* <label for="name">Name</label>
* <input type="text" id="name"
* class="form-control">
* </div>
* <div class="row">
* <div class="col-sm-2 form-group">
* <label for="amount">Amount</label>
* <input type="number" id="amount" class="form-control">
* </div>
* </div>
* <div class="row">
* <div class="col-xs-12">
* <button class="btn btn-success" type="submit">Add</button>
* <button class="btn btn-danger" type="button">Delete</button>
* <button class="btn btn-primary" type="button">Clear</button>
* </div>
* </div>
* </form>
* </div>
* </div>
* In browser we can see the same output

