توسعه یک shopping cart با استفاده از معماری MVC

علی عسگری

تیر 1401

عنوان: توسعه یک shopping cart با استفاده از معماری MVC

نگارش : علی عسگری

نام درس : معماری نرم افزار

استاد درس : دکتر مصطفی فخر احمد

بیان ابزار های مورد استفاده در پروژه :

در این پروژه از معماری مدل ویو کنترلر استفاده شده است ، همچنین از زبان جاوااسکریپت و node.js برای پیاده سازی منطق و back end پروژه استفاده شده است. پروژه حالت server side rendering دارد و از handlebars js برای پیاده سازی بخش view ها و تمپلیت ها و در واقع نمای ظاهری پروژه استفاده شده است.
پروژه دارای یک سیستم حافظه موقت است که از لحظه ران شدن تا قطع اجرا تمامی فعالیت های مورد نیاز را ذخیره و مدیریت می کند.

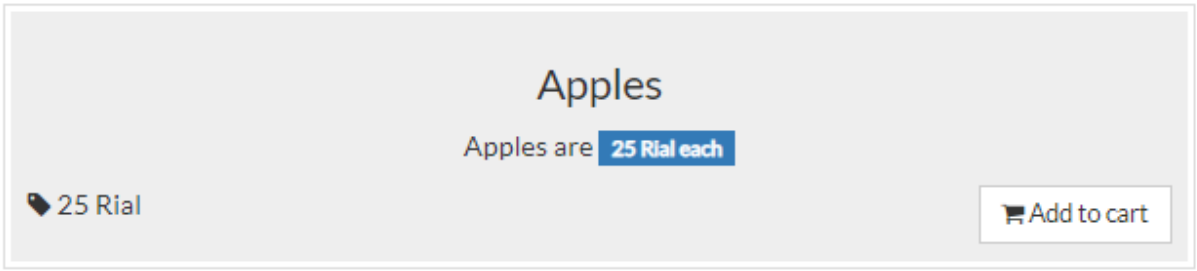
بیان اجمالی کارکرد پروژه :

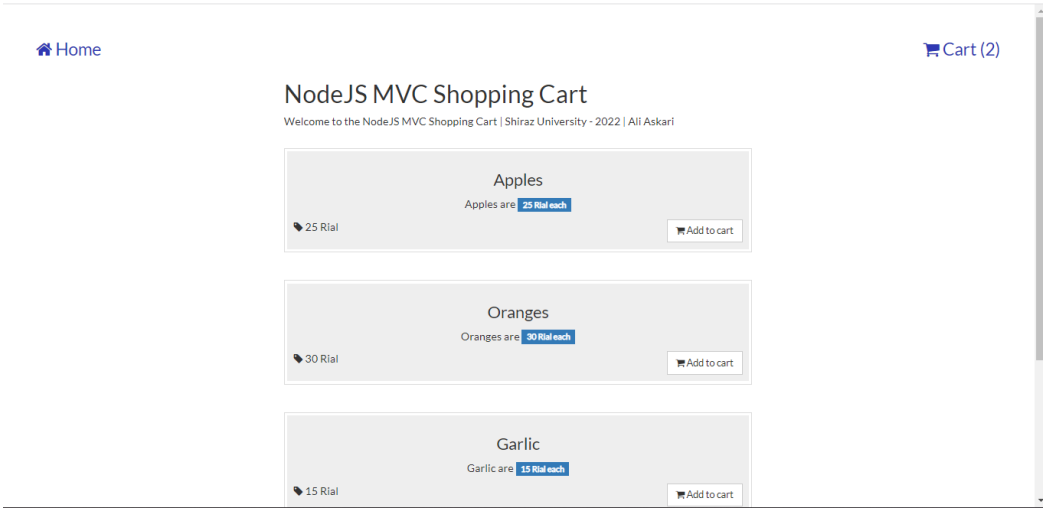
در تصویر زیر فایل های اصلی پروژه که باعث شده اند معماری MVC شکل گیرد را مشاهده میکنید.

لازم به ذکر است که پروژه فایل های دیگری هم دارد که مورد نیاز اند ، برای مثال برای راه اندازی سرور و یا نام گذاری route ها ، اما در کل معماری MVC پیاده سازی شده است و قابلیت اجرایی شدن دارد.

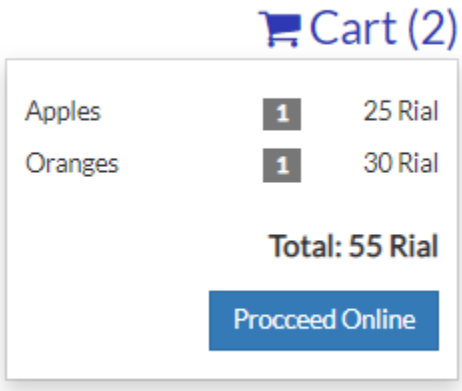
اجرای پروژه :

تصویر یک کارت خرید را مشاهده میکنید.

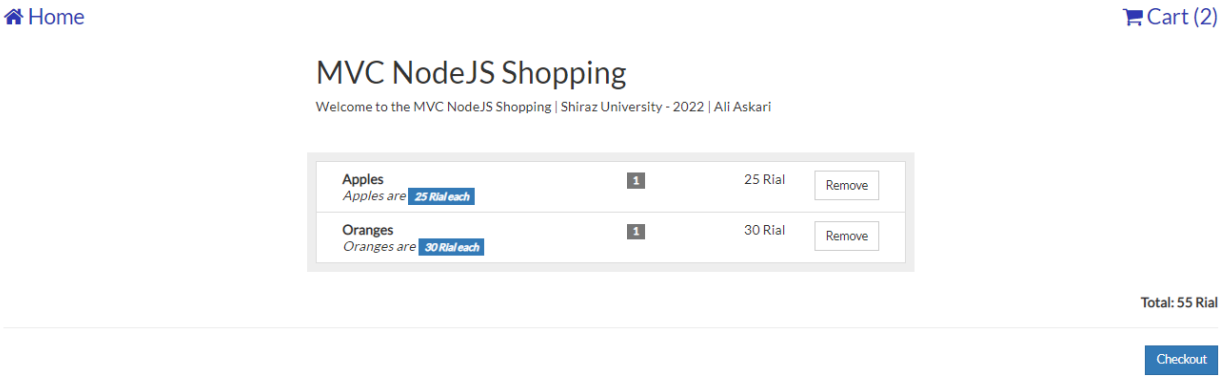




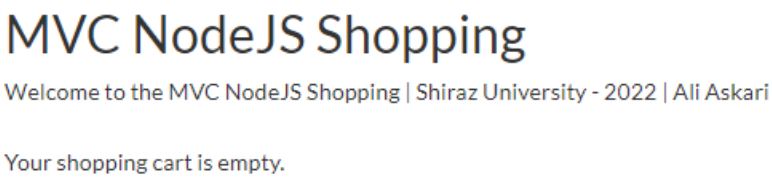
به تصویر بالا نگاه کنید ، دو محصول پس از کلیک بر روی دکمه طراحی شده به سبد خرید اضافه شده اند. حال بر روی عبارت **CART** که بالا سمت راست قرار دارد کلیک میکنیم و با تصویر زیر مواجه می شویم.



حال در ادامه بر روی دکمه ادامه کلیک میکنیم و به صفحه بعدی که نمایش سبد خرید هست میرسیم.



تصویر بالا دکمه نهایی حذف یک محصول انتخاب شده را دارا هست ، در صورت حذف هر دو محصول با پیغامی مشابه تصویر زیر مواجه می شویم.



مدل :

وظیفه مدیریت دیتا و ایجاد تغییر بر روی دیتا را بر عهده گرفته است ، به تصویر زیر دقت کنید.

```
module.exports = function Cart(cart) {
  this.items = cart.items || {};
  this.totalItems = cart.totalItems || 0;
  this.totalPrice = cart.totalPrice || 0;

  this.add = function (item, id) {
    var cartItem = this.items[id];
    if (!cartItem) {
      cartItem = this.items[id] = { item: item, quantity: 0, price: 0 };
    }
    cartItem.quantity++;
    cartItem.price = cartItem.item.price * cartItem.quantity;
    this.totalItems++;
    this.totalPrice += cartItem.item.price;
  };

  this.remove = function (id) {
    this.totalItems -= this.items[id].quantity;
    this.totalPrice -= this.items[id].price;
    delete this.items[id];
  };

  this.getItems = function () {
    var arr = [];
    for (var id in this.items) {
      arr.push(this.items[id]);
    }
    return arr;
  };
};
```

فانکشن های اصلی مربوط به اضافه کردن یک محصول به سبد خرید و حذف محصول از سبد خرید در این فایل تحت عنوان model و cart.js قرار دارد.

کنترلر :

بحث کنترلر واسطه بودن بین view و model است ، یعنی کنترلر تعیین میکند در کدام بخش view کدام اطلاعات نمایش داده شوند و یا کدام تغییرات بر روی دیتا ، برای کدام بخش از view انجام شود. به طور ساده فانکشن هایی که برای ایجاد و مدیریت تغییرات در model ایجاد کردیم ، حالا در controller باید برای بخش مناسب view صدا زده شوند تا view بتواند نقش خود را در نمایش و دریافت اطلاعات ایفا کند.

```
var Cart = require("../models/cart");

class controller {
  addContoller(req, res, next) {
    var productId = req.params.id;
    var cart = new Cart(req.session.cart ? req.session.cart : {});
    console.log(cart);
    var product = products.filter(function (item) {
      console.log(item.id);

      return item.id == productId;
    });

    cart.add(product[0], productId);
    req.session.cart = cart;
    res.redirect("/");
  }
  cartContorller(req, res, next) {
    if (!req.session.cart) {
      return res.render("cart", {
        products: null
      });
    }
    var cart = new Cart(req.session.cart);
    res.render("cart", {
      title: "MVC NodeJS Shopping ",
      products: cart.getItems(),
      totalPrice: cart.totalPrice
    });
  }
  removeController(req, res, next) {
    var productId = req.params.id;
    var cart = new Cart(req.session.cart ? req.session.cart : {});

    cart.remove(productId);
    req.session.cart = cart;
    res.redirect("/cart");
  }
}
```

ویو :

کد های مربوط به کلاینت و نمایش دیتا هستند و احساس میکنم به توضیحات زیادی احتیاج نیست.

```
<div class="cart page ">
      <ul class="list-group center col">
          {{# each products }}
          <li class="list-group-item clearfix ">
              <div class="col-xs-5">
                  <strong>{{ this.item.title }}</strong><br/>
                  <em>{{{this.item.description}}}</em>
              </div>
              <div class="col-xs-2 text-right">
                  <span class="badge">{{ this.quantity }}</span>
              </div>
              <div class="col-xs-3 text-right">
                  {{ this.price }} Rial
              </div>
              <div class="col-xs-2 text-right">
                  <a href="/remove/{{this.item.id}}" class="btn btn-default"></i> Remove</a>
              </div>
          </li>
          {{/each}}
      </ul>
  </div>
```

نکات پایانی :

برای تعیین نام route یا URL ها و همچنین نسبت دادن controller مربوط به هر Route ، از کدهای زیر استفاده شده است.

```
router.get("/", function (req, res, next) {
  res.render("index", {
    title: "NodeJS MVC Shopping Cart",
    products: products
  });
});

router.get("/add/:id", Controller.addContoller);
router.get("/cart", Controller.cartContorller);
router.get("/remove/:id", Controller.removeController);
```

همچنین دیتای مربوط به این پروژه فرمت json داشته است و از دیتای های آماده بر بستر اینترنت استفاده شده است .

با احترام و تشکر به خاطر وقتی که برای مطالعه گذاشتید
همچنین از کارکردِ سیستم ویدیوی کوتاهی در فایل پروژه قرار گرفته است
علی عسگری - تیر 1401