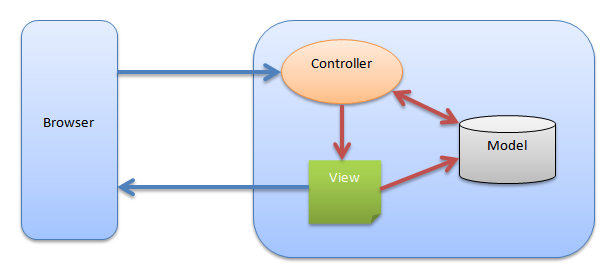
Phần II: GIẢI QUYẾT VẤN ĐỀ

Để thực hiện đồ án này, nhóm HKT đã sử dụng mô hình MVC để xây dựng website này.

Mô hình MVC là một mô hình lập trình web được sử dụng rộng rãi hiện nay, được viết tắt bởi ba từ **Model – View – Controller**, tương ứng với ba bộ phận tạo nên cấu trúc của ứng dụng này.

1. **View -** Thành phần này được tạo bởi dữ liệu thu thập từ dữ liệu mô hình, và giúp người dùng có cái nhìn trực quan về trang web, cũng như ứng dụng.
2. **Model -** Model chịu trách nhiệm cho các thao tác dữ liệu giữa Controller hoặc bất kỳ logic nghiệp vụ liên quan nào khác như: cho phép xem, truy xuất dữ liệu,…
3. **Controller -** Controller xử lý dữ liệu đầu vào từ bàn phím và chuột của người dùng sau đó thông báo tới View và Model. Sau đó liền gửi các lệnh tới Model để thay đổi trạng thái của Mode



Đồ án của nhóm HKT bao gồm những chức năng như:

|  |  |
| --- | --- |
| Tên chức năng | Người phụ trách |
| Đăng nhập, Đăng Xuất | Nguyễn Tấn Tài |
| Đăng ký | Đỗ Gia Huy |
| Tìm kiếm | Đặng Tuấn Kiệt |
| Phân trang | Đặng Tuấn Kiệt |
| Thêm, xóa , sửa giỏ hàng | Đỗ Gia Huy |
| Thêm, sửa, xóa admin (tài khoản) | Đặng Tuấn Kiệt |
| Show hoa don, xem chi tiet (hóa đơn) | Nguyễn Tấn Tài |
| Thêm, sửa , xóa admin (sản phẩm) | Đỗ Gia Huy |
| Thiết kế và đổ dữ liệu lên profile | Đặng Tuấn Kiệt |
| Cập nhật mật khẩu profile | Đặng Tuấn Kiệt |
| Cập nhật thông tin profile | Đỗ Gia Huy |
| Hiển thị danh sách các hóa đã mua của user | Nguyễn Tấn Tài |

Phần của Tấn Tài

**Đăng nhập**

Bước 1: muốn bảo mật thông tin của người dùng tránh bị hack thông tin Username và Password

* Cài đặt thư viện **csrf và đọc document để biết cách sử dụng thư viện**
* Vị trí nạp thư viện **: me.js ở thư mục routes**

const meController = require('../app/controllers/MeController');

const csrf = require('csurf');

const csrfProtection = csrf();//bảo mật thông tin

router.get("/login",csrfProtection,notLoggedIn,meController.login);

router.post("/login",meController.xlogin);

Bước 2: Tạo một folder controler và tại file MeController

* Bước 1: cài đặt thư viện **passport** và đọc document để biết cách sử dụng thư viện

const passport = require('passport');//thư viện passport

Khai báo phần login: nơi xử lý login với phương thức GET

login(req,res,next) {

   res.render("login",{csrfToken: req.csrfToken()});

}

Ta render ra giao diện login và chuyển thêm req.csrfToken do thư viện **csrf** tự động tại sau khi ta submit form

Khai báo phần xlogin: nơi xử lý login với phương thức POST

xlogin(req,res,next) {

  passport.authenticate('local',{

  successRedirect:'/me/profile',

  failureRedirect:'/me/login',

  failureFlash: true,

})(req,res,next);

}

passport: là thư viện đã được khai báo.

authenticate: là option của thư viện passport

Phần của Tuấn Kiệt

**CHỨC NĂNG TÌM KIẾM**

**Bước 1: tạo một file site.js trong folder routes**

* router.get('/search', siteController.search);

khai báo router vói phương thức get với path: ‘/search’

* Tạo file SiteController trong folder controller và khai báo search

  search(req,res,next) { //{1}

    var str = req.query.key; //{2}

    //Tìm kiếm dữ liệu trong database theo key

    Product.find({name: { $regex: str, $options: 'i' }})//{3}

      .then((product) => { //{4}

        res.render('search', { //{5}

          product: mutipleMongooseToObject(product), //{6}

        });

      })

      .catch(next); //{7}

  }

**{1}: khai báo option controller search và truyền 3 tham số req, res, next**

**{2}: khai báo biến str để lưu trữ req.query.key -> lấy từ giá trị được bên phía client**

**{3}:** const Product = require('../models/products');

**->** Product là model được khai báo với option find từ thư viện hỗ trợ của mongoose. Tìm theo tên trong collection product với toán tử tìm kiếm $regex với $options: ‘i’ là kiểu tìm kiếm gần giống nhất.

{4}: .then để nhận lại dữ liệu từ Product.find và dữ liệu này được nhận lưu trữ ở product.

{5}:render file search.hbs -> 'search', { truyền dữ liệu đã bắt được } vì hbs không cho phép chúng ta truy cập các thành phần của một đối tượng, nên ta phải truy cập vào các thành phần của một đối tượng bằng các thông qua việc tạo option với mongoose -> mutipleMongooseToObject mục đích làm điều này là để chuyển object của mongodb thành 1 array để có thể truy xuất hướng đội tượng.

{6}: .catch là để bắt lỗi và gọi function next và trả lỗi về