TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

--------- o0o --------

BÁO CÁO ĐỒ ÁN III

Đề tài: Tìm hiểu về NodeJS một nền tảng Server side được xây dựng dựa trên Javascript Engine (V8 Engine) và ứng dụng trong xây dựng website thương mại điện tử.

Giảng viên: TS. Trần Vĩnh Đức

Sinh viên thực hiện:

Nguyễn Đình Quang – MSSV:20146574

Hà Nội, tháng 05, năm 2017

Mục lục

Phần I: Lời mở đầu.

Hiện nay có rất nhiều các công nghệ phát triển web như PHP, ASP, JSP, Node.js,...Mỗi công nghệ sẽ có những ưu nhược điểm riêng trong đó Node.js đang là một xu hướng mới cho các lập trình viên theo đuổi. Node.js là một nền tảng chạy trên môi trường V8 JavaScript runtime - một trình thông dịch JavaScript cực nhanh chạy trên trình duyệt Chrome. Nodejs sử dụng rộng bởi hàng ngàn lập trình viên trên toàn thế giới. NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ Windows cho tới Linux, MacOs nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất. Node.js sử dụng một mô hình luồng duy nhất với sự kiện lặp. cơ chế tổ chức sự kiện giúp các máy chủ để đáp ứng một cách không ngăn chặn và làm cho máy chủ cao khả năng mở rộng như trái ngược với các máy chủ truyền thống mà tạo đề hạn chế để xử lý yêu cầu. Node.js sử dụng một chương trình đơn luồng và các chương trình tương tự có thể cung cấp dịch vụ cho một số lượng lớn hơn nhiều so với yêu cầu máy chủ truyền thống như Apache HTTP Server.

Phần II: Nội dung.

I: Tổng quan về Node.js

1. Giới thiệu chung.

Javascript ngày càng trở nên phổ biến hơn với nhiều tính năng và các thư viện được hỗ trỡ cho developer, điều đó khiến cho các giao diện web càng trở nên sinh động hơn. Mọi thứ mà chúng ta có thể làm được trên web ngày nay là Javascript có thể chạy được trên server, cũng như chạy được trên browser, điều này là khó tưởng tượng trong những năm trở lại đây, hoặc nó chỉ đóng gói trong môi trường sandboxed như Flash hoặc JavaApplets. Thật vậy, rõ ràng rằng từ xưa đến nay chúng ta vẫn quan niệm rằng lập trình bên phía server chỉ dùng được những ngôn ngữ như php, ruby, ... nhưng từ khi node js ra đời nó mang đến tư tưởng mới cho việc lập trình cả bên phía server cũng như bên phía client. Và đối với node js thì chúng ta biến những điều mà trước kia chỉ có thể thao tác được bên phía client thì nay cũng thao tác và xử lý được trên server, và ngược lại. Node.js được viết bằng ngôn ngữ javascript, nó là một trình biên đóng gói của Google’s V8 JavaScript engine, libuv platform abstraction layer, và một thư viện lõi được viết bằng Javascript. Mục tiêu của Node js là làm cho web có khả năng push như trong một số ứng dụng gmail. Node js cung cấp công cụ giúp lập trình viên có thể làm việc trong non-blocking, mô hình I/O . Sau hơn 20 năm nghiên cứu, xây dựng và phát triển, nhóm kĩ sư đã cho ra đời sản phẩm ứng dụng web node js chạy thời gian thực và kết nối 2 chiều client và server, cho phép trao đổi dữ liệu một cách tự do.

2. Cách thức hoạt động.

Ý tưởng chính của Node js là sử dụng non-blocking, hướng sự vào ra dữ liệu thông qua các tác vụ thời gian thực một cách nhanh chóng. Bởi vì, Node js có khả năng mở rộng nhanh chóng, khả năng xử lý một số lượng lớn các kết nối đồng thời bằng thông lượng cao. Nếu như các ứng dụng web truyền thống, các request tạo ra một luồng xử lý yêu cầu mới và chiếm RAM của hệ thống thì việc tài nguyên của hệ thống sẽ được sử dụng không hiệu quả. Chính vì lẽ đó giải pháp mà Node js đưa ra là sử dụng luồng đơn (Single-

Threaded), kết hợp với non-blocking I/O để thực thi các request, cho phép hỗ trợ hàng chục ngàn kết nối đồng thời.

Cơ chế của Node js Application xử lý Model như sau:

- Client gửi request đến Web Server
- Node js Web Service duy trì trong nội bộ một luồng giới hạn để cung cấp dịch vụ cho Client Request.
- Node js Web Service nhận tất cả các request và đặt chúng vào một trong Queue. Nó được gọi là một Event Queue.
- Node js Web Service nội bộ có một thành phần được gọi là "Event Loop".
- Event Loop chỉ sử dụng một luồng đơn để xử lý Model.
- Event Loop kiểm tra tất cả các Request đặt trong Event Queue. Nếu không có request nào thì chờ request đến vô thời hạn
- Nếu có request thì sẽ lấy một request từ Event Queue:

- Khởi động quá trình xử lý tiến trình từ client request
- Nếu Client Request không chứa nhiều Blocking I/O thì xử lý tất cả mọi thứ và chuẩn bị cho quá trình gửi lại phản hồi cho phía client.
- Nếu Client Request chứa nhiều Blocking I/O như việc tương tác với cơ sở dữ liệu, tập tin hệ thống, dịch vụ mở rộng, thì nó sẽ thực hiện theo các phương pháp tiếp cận khác nhau.

  + Kiểm tra các luồng sẵn có từ nội bộ bên trong của request gửi lên

  + Chọn một luồng và chỉ định cho client request tương ứng với luồng đó

  + Luồng đó phải có trách nhiệm với reuqest đó, xử lý nó, thực thi các hoạt động Blocking I/O, chuẩn bị các phản hồi và gửi lại cho Event Loop.

  + Event Loop gửi lại phản hồi tương ứng cho client.

3. NPM: The Node Package Manager.

Khi thảo luận về Node js thì một điều chắc chắn không nên bỏ qua là xây dựng package quản lý sử dụng các cộng cụ NPM mà mặc định với mọi cài đặt Node js. Ý tưởng của mô-đun NPM là khá tương tự như Ruby-Gems: một tập hợp các hàm có sẵn có thể sử dụng được, thành phần tái sử dụng, tập hợp các cài đặt dễ dàng thông qua kho lưu trữ trực tuyến với các phiên bản quản lý khác nhau.

Danh sách các mô-đun có thể tìm trên web NPM package hoặc có thể truy cập bằng cách sử dụng công cụ NPM CLI sẽ tự động cài đặt với Node js.

Một số các module NPM phổ biến nhất hiện nay là:

- expressjs.com/ - Express.js, một Sinatra-inspired web framework khá phát triển của Node.js, chứa rất nhiều các ứng dụng chuẩn của Node.js ngày nay.

- connect - Connect là một mở rộng của HTTP server framework cho Node.js, cung cấp một bộ sưu tập của hiệu suất cao "plugins" được biết đến như là trung gian; phục vụ như một nền tảng cơ sở cho Express
- socket.io and sockjs - Hai thành phần Server-side websockets components nổi tiếng nhất hiện nay.
- Jade - Một trong những engines mẫu, lấy cảm hứng từ HAML, một phần mặc định trong Express.js.
- mongo and mongojs - MongoDB hàm bao để cung cấp các API cho cơ sở dữ liệu đối tượng trong MongoDB Node.js
- redis - thư viện Redis client.
- coffee-script - CoffeeScript trình biên dịch cho phép developers viết các chương trình Node.js của họ dùng Coffee.
- underscore (lodash, lazy) - Thư viện tiện ích phổ biến nhất trong JavaScript, package được sử dụng với Node.js, cũng như hai đối tác của mình, hứa hẹn hiệu suất tốt hơn bằng cách lấy một cách tiếp cận thực hiện hơi khác nhau.
- forever - Có lẽ là tiện ích phổ biến nhất để đảm bảo rằng một kịch bản nút cho chạy liên tục. Giữ quá trình Node.js của bạn lên trong sản xuất đối mặt với bất kỳ thất bại không ngờ tới.

II: Nội dung chính.

1. Website thương mại điện tử.

Ngày nay việc mua bán trực tuyến đã trở lên vô cùng thông dụng trên toàn thế giới. Website thương mại điện tử là trang thông tin điện tử được thiết lập để phục vụ một phần hoặc toàn bộ quy trình của hoạt động mua bán hàng hóa hay cung ứng dịch vụ, từ trưng bày giới thiệu hàng hóa đến giao kết hợp đồng, cung ứng dịch vụ, thanh toán và dịch vụ sau bán hàng.

Website thương mại điện tử chính là một website bán hàng, là kênh thông tin giới thiệu về doanh nghiệp, của hàng đồng thời là kênh quảng bá hữu hiệu. Đa tính năng như vậy nên website thương mại điện tử đã trở thành yếu tố không thể thiếu đối với bất kỳ công ty, đơn vị kinh doanh nào nếu không muốn bị tụt lại so với đối thủ trong chặng đua khắc nghiệt!

2. Ví dụ một số website thương mại điện tử hàng đầu việt nam.

+ Lazada.vn

Lazada là website mua sắm trực tuyến nhiều người sử dụng nhất hiện nay tại Việt Nam. Tuy nhiên công ty Lazada không phải của Việt Nam mà là của công ty Singapore có chi nhánh tại Việt Nam mà thôi. Lazada là công ty bán hàng online lớn nhất tại khu vực Đông Nam Á này với nhiều chi nhánh tại các nước lớn khu vực như Indonesia, Thailand, Philippines, Malaysia, Singapore. Nhưng mà công ty này không cung cấp tất cả hàng hóa dịch vụ mà chủ yếu là tạo ra

sàn giao dịch online cho các cửa hàng đăng ký bán hàng trên website, công ty sẽ đảm bảo về giao dịch trực tuyến và quản lý cửa hàng, khách hàng.

+ Sendo.vn

Sendo.vn là chợ trực tuyến của Tập đoàn FPT kết nối người mua và người bán trên toàn quốc. Xuất thân là một dự án Thương mại Điện tử do Công ty CP Dịch vụ Trực tuyến FPT (FPT Online) phát triển, Sendo.vn ra mắt người tiêu dùng vào tháng 9/2012. Ngày 13/5/2014, Công ty CP Công nghệ Sen Đỏ được thành lập, trực thuộc Tập đoàn FPT, là đơn vị chủ quản Sendo.vn.

3. Phân tích thiết kế và ứng dụng nền tảng Node.js cũng như các framework vào việc xây dựng website bán hàng.

3.1: Expressjs Framework

3.1.1: Framework giúp cho việc phát triển ứng dụng được rút ngắn đi rất nhiều. Cũng như các framework dựa trên những ngôn ngữ khác như Rails (Ruby); Django (Python); Laravel, CakePHP (PHP)... Express được xây dựng dựa trên Node.js.

Express hỗ trợ việ phát triển ứng dụng web theo mô hình MVC. Mô hình phổ biến cho việc lập trình web hiện nay.

Cho phép định nghĩa Middleware hỗ trợ cho việc tổ chức và tái sử dụng code.

Định nghĩa routes và cá request method đến server một cách dễ dàng.

3.1.2: Cài đặt.

Giả sử đã cài đặt trên máy tính Node.js, tạo thư mục ứng dụng website và khởi tạo project.

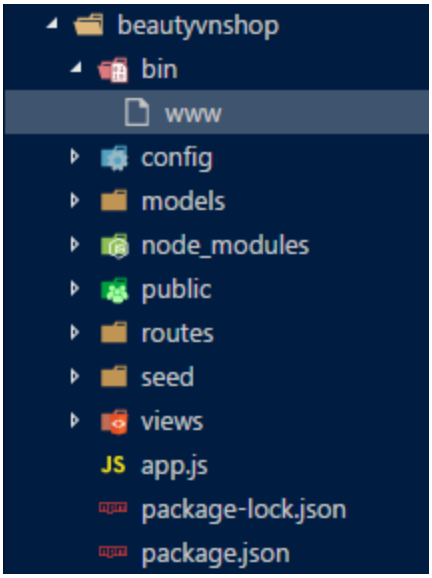
```
$ mkdir myapp
$ cd myapp
```

```
$ npm init
```

Tiếp theo tiến hành cài đặt Express.

```
$ npm install express --save
```

Cấu trúc thư mục express.



Khi khởi tạo một Web Application với ExpressJS, chúng ta sẽ có một đối tượng đại diện cho Web App đó, thường được gán với biến app. Đối tượng này có thể khai báo các middleware thông qua các hàm : app.use() hoặc app.METHOD (trong đó METHOD sẽ là cá kiểu HTTP Method được ExpressJS hỗ trợ, dưới dạng tên là chữ viết thường, vd app.get(), app.post()).

Ví dụ dưới đây mô tả một hàm ko khai báo đường dẫn cụ thể, do đó nó sẽ được thực hiện mỗi lần request:

```
var app = express()

app.use(function (req, res, next) {
  console.log('Time:', Date.now())
  next()
```

```
})
```

Ví dụ dưới đây dùng hàm use đến đường dẫn /user/:id. Hàm này sẽ được thực hiện mỗi khi request đến đường dẫn /user/:id bất kể phương thức nào (GET, POST,...):

```
app.use('/user/:id', function (req, res, next) {
  console.log('Request Type:', req.method)
  next()
})
```

Tiếp theo là một ví dụ cho hàm được thực hiện mỗi khi truy cập đến đường dẫn /user/:id bằng phương thức GET:

```
app.get('/user/:id', function (req, res, next) {
  res.send('USER')
})
```

Khi muốn gọi một loạt hàm middleware cho một đường dẫn cụ thể, chúng ta có thể thực hiện như ví dụ dưới đây, bằng cách khai báo liên tiếp các tham số là các hàm sau tham số đường dẫn :

```
app.use('/user/:id', function (req, res, next) {
  console.log('Request URL:', req.originalUrl)
  next()
}, function (req, res, next) {
  console.log('Request Type:', req.method)
  next()
})
```

Hoặc chúng ta có thể tách ra thành 2 lần khai báo app.use, gọi là multiple routes, tuy nhiên ở các hàm phía trước cần gọi hàm next() khi kết thúc mỗi hàm, nếu không như ví dụ dưới đây, route thứ 2 sẽ không bao giờ được thực hiện do hàm thứ 2 trong route thứ nhất không gọi đến hàm next():

```
app.get('/user/:id', function (req, res, next) {
  console.log('ID:', req.params.id)
  next()
}, function (req, res, next) {
  res.send('User Info')
})

// handler for the /user/:id path, which prints the user ID
app.get('/user/:id', function (req, res, next) {
  res.end(req.params.id)
})
```

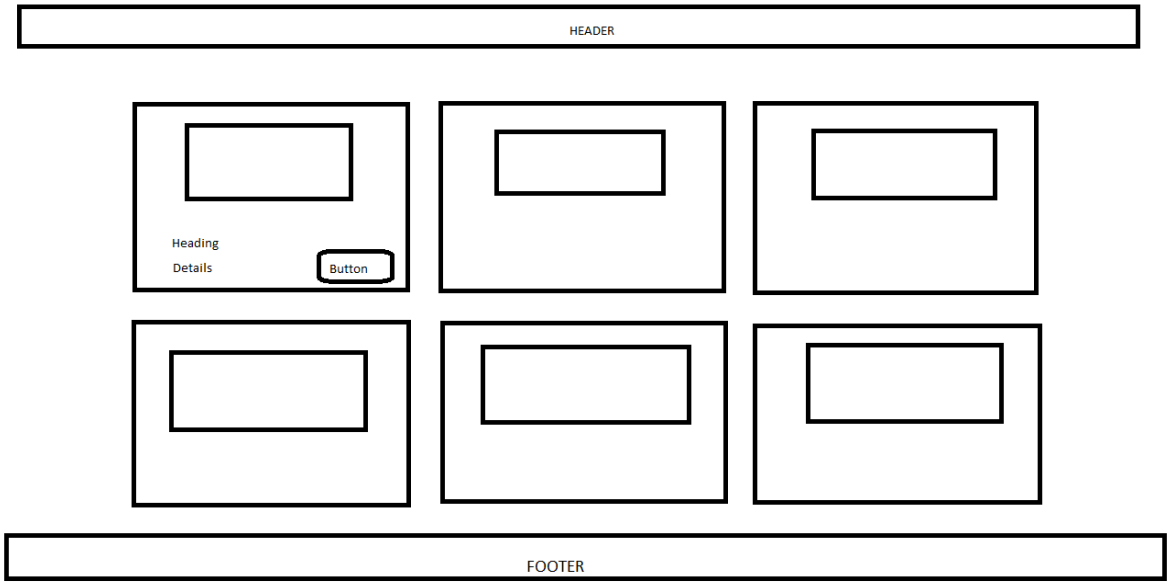
Khi muốn bỏ qua các hàm middleware tiếp theo không thực hiện nữa, chúng ta sẽ sử dụng lệnh next('route'), tuy nhiên việc này chỉ tác dụng vưới các hàm middleware được load thông qua hàm app.METHOD hoặc router.METHOD.

Ví dụ dưới đây mô tả một hàm middleware sẽ kết thúc ngay lập tức khi tham số id=0:

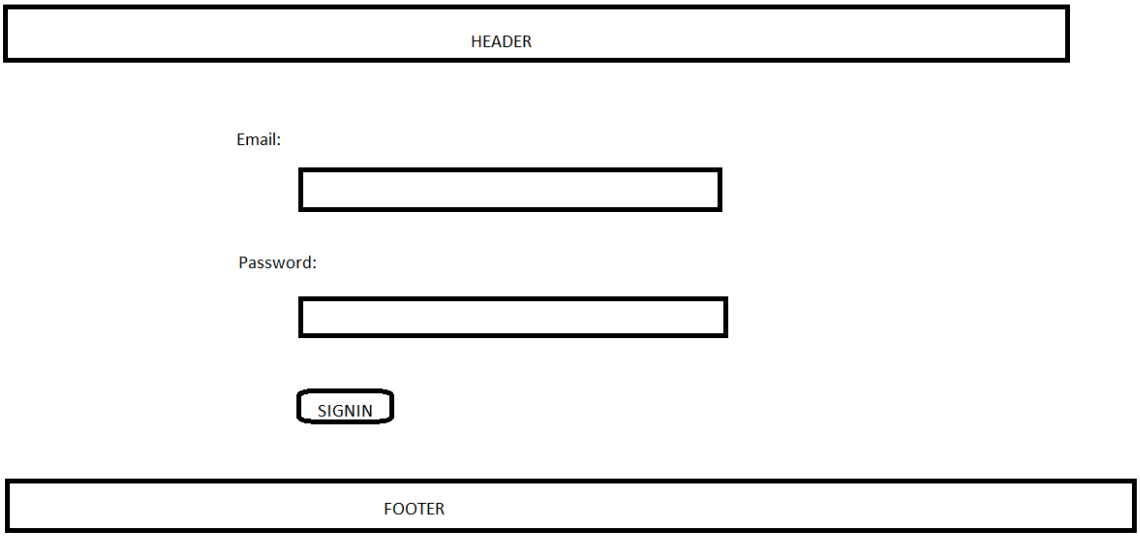
```
app.get('/user/:id', function (req, res, next) {
  // if the user ID is 0, skip to the next route
  if (req.params.id === '0') next('route')
  // otherwise pass the control to the next middleware function in this stack
  else next()
}, function (req, res, next) {
  // render a regular page
  res.render('regular')
})

// handler for the /user/:id path, which renders a special page
app.get('/user/:id', function (req, res, next) {
  res.render('special')
})
```

3.2: Phân tích thiết kế giao diện.

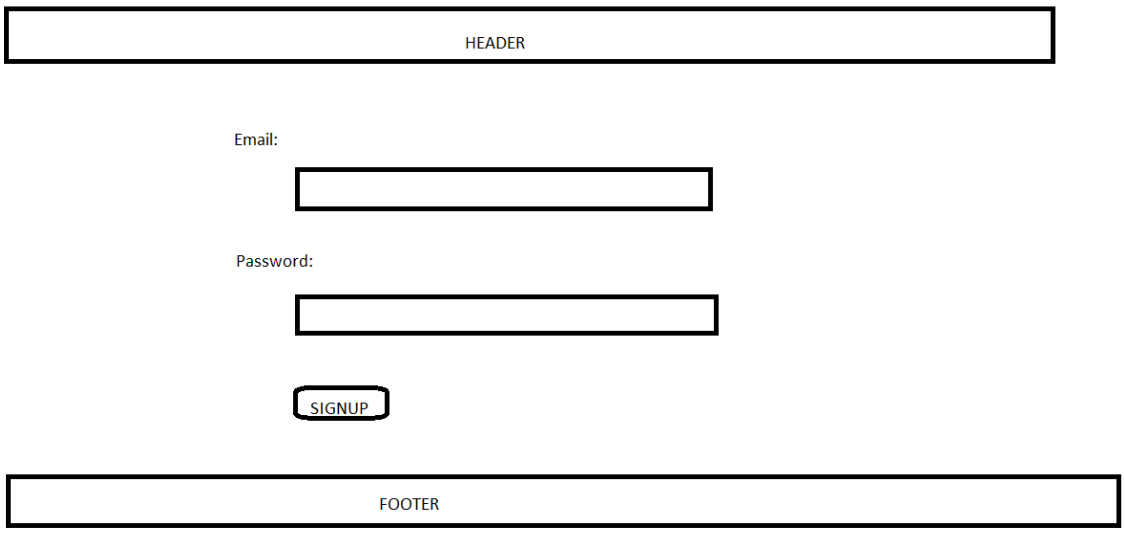
3.2.1: Giao diện chính.



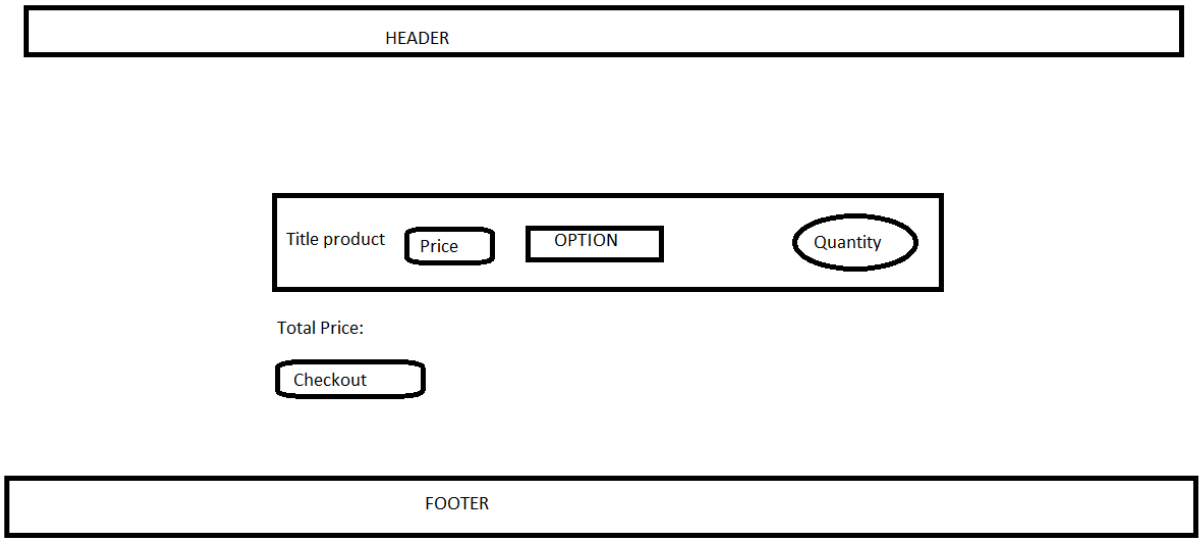
3.2.2: Giao diện đăng nhập.



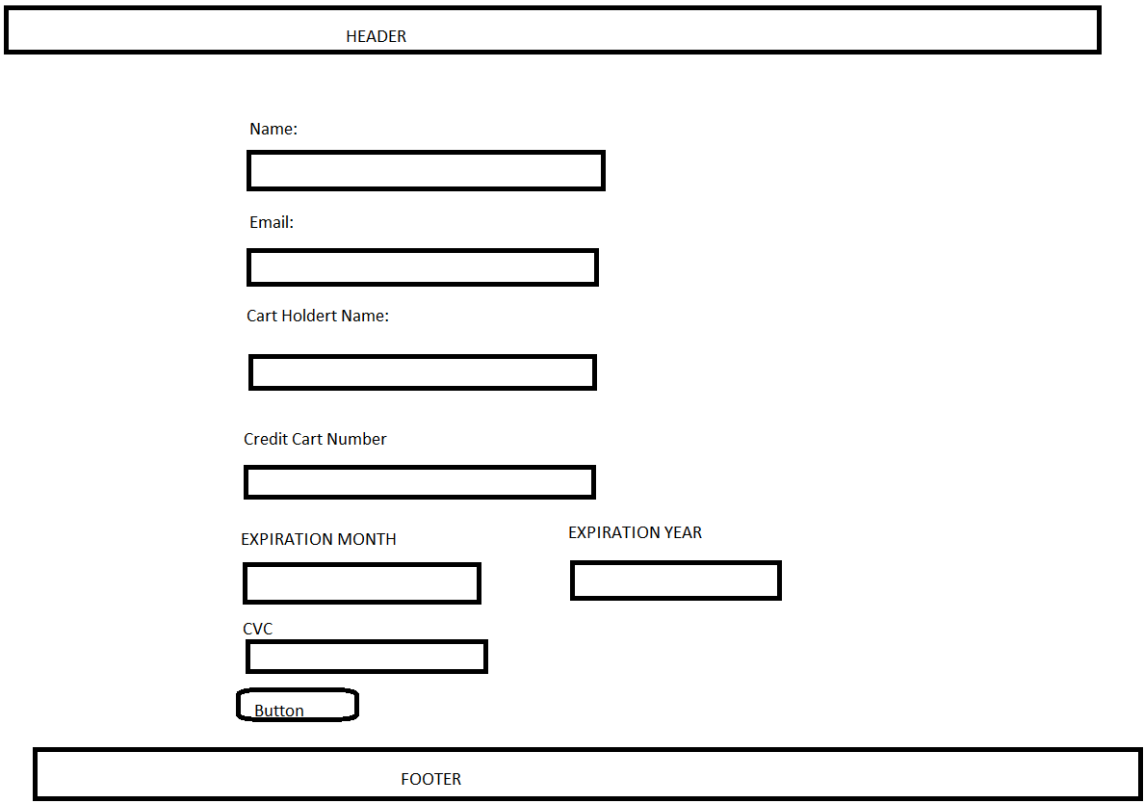
3.2.3: Giao diện đăng kí tài khoản.



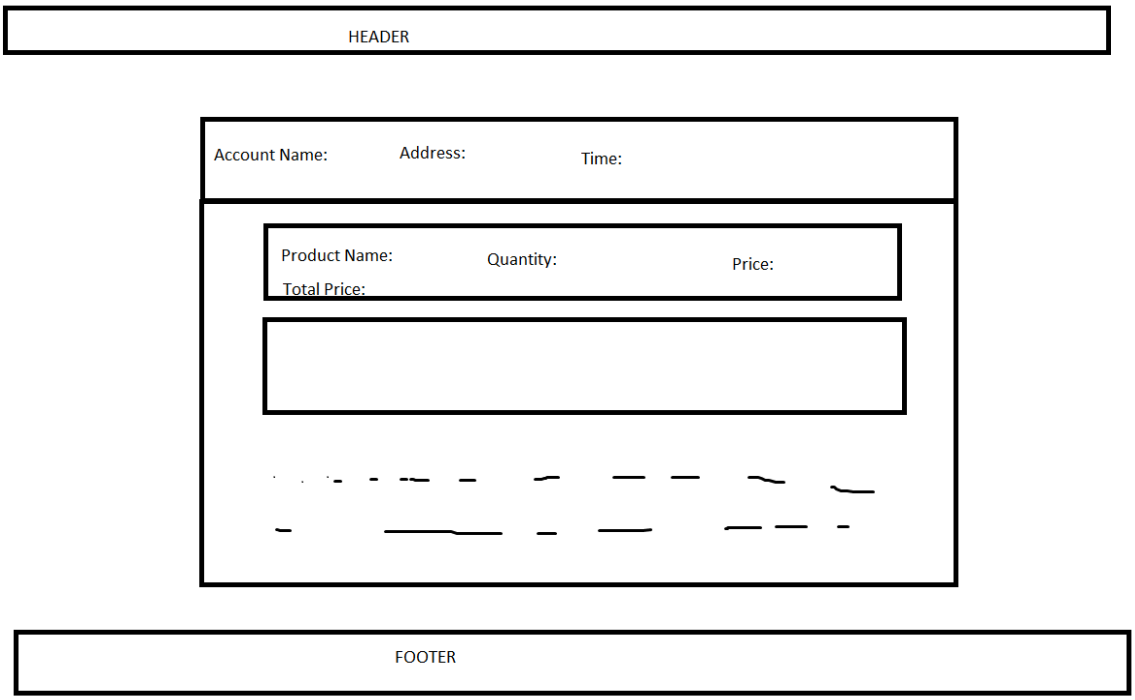
3.2.4: Giỏ hàng sản phẩm.



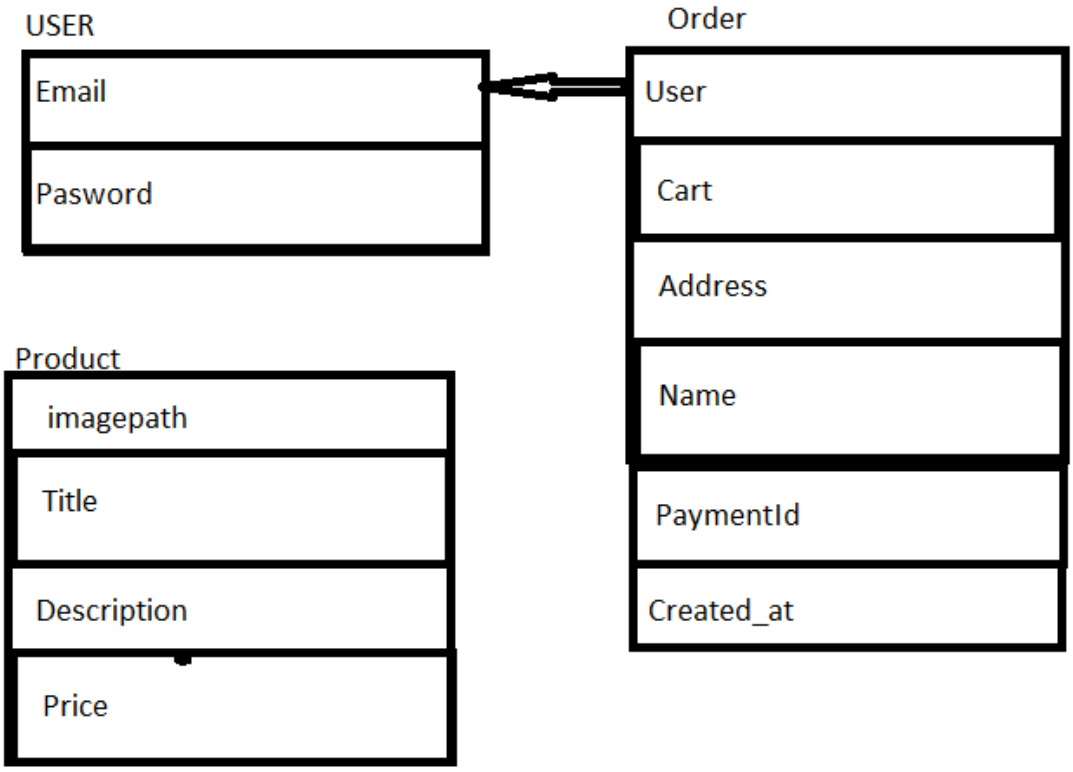
3.2.5: Payment.


HEADER
Name:
Email:
Cart Holdert Name:
Credit Cart Number
EXPIRATION MONTH
EXPIRATION YEAR
CVC
Button
FOOTER

3.2.6: Profile.


HEADER
Account Name:
Address:
Time:
Product Name:
Quantity:
Price:
Total Price:
FOOTER

3.3: Cơ sở dữ liệu.



Với việc số lượng bản ghi ngày càng lớn đòi hỏi việc insert load dữ liệu nhanh chóng thì NoSQL chắc hẳn là một giải pháp an toàn và hợp lý cho các tập đoàn lớn như Facebook, Google.

Đối với website này em sẽ sử dụng MongoDB làm hệ quản trị CSDL cho trang web.

+ Package sử dụng trong project:

- Mongodb
- Mongoose

+) Cài đặt mongose:

```
npm install mongoose
```

+) Kết nối với CSDL:

```
var mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/my_database');
```

+) Khai báo Model:

1. Order

```
var schema = new Schema({
    user: {type: Schema.Types.ObjectId, ref: 'User'},
    cart: {type: Object, required: true},
    address: {type: String, required: true},
    name: {type: String, required: true},
    paymentId: {type: String, required: true},
    created_at: { type: Date, required: true, default: Date.now }
});
```

2. User

```
var userSchema = new Schema({
    email: {type: String, required: true},
    password: {type: String, required: true}
});
```

3. Product

```
var schema = new Schema({
    imagePath: {type: String, required: true},
    title: {type: String, required: true},
    description: {type: String, required: true},
    price: {type: Number, required: true}
});
```

Ví dụ: CSDL được lưu dưới dạng file JSON.

VD1: users

{
  "_id": {
    "$oid": "593822df06567228484098f7"
  },
  "password": "$2a$05$E49acb.qbMwPk.2iW5Xiz.oxSTkjJnMqr5Ycs54esJF3adtEe5ZY6",
  "email": "hoamoclan280296@gmail.com",
  "__v": 0
}

VD2: orders


```
{
   "_id": {
      "$oid": "59382365065672284840 98f8"
   },
   "user": {
      "$oid": "593822df06567228484098f7"
   },
   "cart": {
      "items": {
         "59381ecd133d7a1af81eb7a3": {
            "item": {
               "_id": "59381ecd133d7a1af81eb7a3",
               "imagePath": "https://images-na.ssl-images-amazon.com/images/I/51Cajbvpf9L.jpg",
               "title": "History",
               "description": "Promise Me Forever Debbie Macomber Classics",
               "price": 18,
               "__v": 0
            },
            "qty": 1,
            "price": 18
         },
         "59381ecd133d7a1af81eb7a4": {
            "item": {
               "_id": "59381ecd133d7a1af81eb7a4",
               "imagePath": "https://images-na.ssl-images-amazon.com/images/I/51ZewuY2UqL._SY346_.jpg",
               "title": "Romance",
```

```
            "description": "Come Sundown Nora Roberts",
            "price": 15,
            "__v": 0
          },
          "qty": 2,
          "price": 30
        },
        "59381ecd133d7a1af81eb7a5": {
          "item": {
            "_id": "59381ecd133d7a1af81eb7a5",
            "imagePath": "https://images-na.ssl-images-amazon.com/images/I/510KmiWD3iL._SY346_.jpg",
            "title": "ML",
            "description": "Bayes Theorem: A Visual Introduction",
            "price": 25,
            "__v": 0
          },
          "qty": 1,
          "price": 25
        }
      },
      "totalQty": 4,
      "totalPrice": 73
    },
    "address": "Ỷ Đốn",
    "name": "Nguyễn Đình Quang",
    "paymentId": "ch_1AS6bMBkQ1C3wsaYUaamFkwk",
    "created_at": {
      "$date": "2017-06-07T16:01:41.206Z"
```

},

"__v": 0

}

VD3: products

{

"_id": {

"$oid": "59381ecd133d7a1af81eb7a3"

},

"imagePath": "https://images-na.ssl-images-amazon.com/images/I/51Cajbvpf9L.jpg",

"title": "History",

"description": "Promise Me Forever Debbie Macomber Classics",

"price": 18,

"__v": 0

}

3.4: Sử lý đăng nhập, đăng kí tài khoản.

Đối với việc chứng thực tài khoản thì pasport.js là module phổ biến nhất của node.js. Nó được thiết kế là một middleware hết sức linh hoạt cho bạn khả năng tùy biến cao với rất nhiều các kịch bản authentication: bạn có thể sử dụng Twitter, Facebook, Google thậm chí là qua username-password trong database. Cũng có thể tùy biến chính xác các route nào cần phải authentication.

Các bước sử dụng:

1. Require model pasport và sử dụng passport.initialize() và passport.session() với express

```
app.use(session({
  secret: 'mysupersecret',
  resave: true,
  saveUninitialized: true,
  store: new MongoStore({ mongooseConnection: mongoose.connection }),
  cookie: { maxAge: 30 * 60 * 1000 }
}));
app.use(passport.initialize());
app.use(passport.session());

app.use(function(req, res, next) {
    res.locals.login = req.isAuthenticated();
```

```
    res.locals.session = req.session;
    next();
});

```

2. Cấu hình kịch bản cho Pasport và thiết lập 2 hàm serializeUser, deserializeUser.

```
var passport = require('passport');
var User = require('../models/user');
var LocalStrategy = require('passport-local').Strategy;
// var FacebookStrategy = require('passport-facebook').Strategy;

passport.serializeUser(function(user, done) {
  done(null, user.id);
});

passport.deserializeUser(function(id, done) {
  User.findById(id, function(err, user) {
    done(err, user);
  });
});

passport.use('local.signup', new LocalStrategy({
  usernameField: 'email',
  passwordField: 'password',
  passReqToCallback: true
}, function(req, email, password, done) {
  req.checkBody('email', 'Invalisd Email!').notEmpty().isEmail();
  req.checkBody('password', 'Invalid password!').notEmpty().isLength({min:
4});

  var errors = req.validationErrors();
  if (errors) {
    var messages = [];
    errors.forEach(function(error) {
      messages.push(error.msg);
    });
    return done(null, false, req.flash('error', messages));
  }
  User.findOne({
    'email': email
  }, function(err, user) {
    if (err) {
      return done(err);
    }
```

```
      if (user) {
        return done(null, false, {
          message: 'Email is already in use!'
        });
      }
      var newUser = new User();
      newUser.email = email;
      newUser.password = newUser.encryptPassword(password);
      newUser.save(function(err, result) {
        if (err) {
          return done(err);
        }
        return done(null, newUser);
      });
    });
  }));

  passport.use('local.signin', new LocalStrategy({
    usernameField: 'email',
    passwordField: 'password',
    passReqToCallback: true
  }, function(req, email, password, done) {
    req.checkBody('email', 'Invalis email !').notEmpty().isEmail();
    req.checkBody('password', 'Invalid password !').notEmpty();
    var errors = req.validationErrors();
    if (errors) {
      var messages = [];
      errors.forEach(function(error) {
        messages.push(error.msg);
      });
      return done(null, false, req.flash('error', messages));
    }
    User.findOne({
      'email': email
    }, function(err, user) {
      if (err) {
        return done(err);
      }
      if (!user) {
        return done(null, false, {
          message: ' No user found!.'
        });
      }
      if (!user.validPassword(password)) {
        return done(null, false, {
```

```
          message: 'Wrong password!'
        });
      }
      return done(null, user);
    });
  }));

```

3.Thiết lập route

```
router.get('/signup', function(req, res, next) {
  var messages = req.flash('error');
  res.render('user/signup', {
    csrfToken: req.csrfToken(),
    messages: messages,
    hasErrors: messages.length > 0
  });
});

router.post('/signup', passport.authenticate('local.signup', {
  failureRedirect: '/user/signup',
  failureFlash: true
}), function(req, res, next) {
  if (req.session.oldUrl) {
    var oldUrl = req.session.oldUrl;
    req.session.oldUrl = null;
    res.redirect(oldUrl);
  } else {
    res.redirect('/user/profile');
  }
});

router.get('/signin', function(req, res, next) {
  var messages = req.flash('error');
  res.render('user/signin', {
    csrfToken: req.csrfToken(),
    messages: messages,
    hasErrors: messages.length > 0
  });
});

router.post('/signin', passport.authenticate('local.signin', {
  failureRedirect: '/user/signin',
  failureFlash: true
}), function(req, res, next) {
```

```
  if (req.session.oldUrl) {
    var oldUrl = req.session.oldUrl;
    req.session.oldUrl = null;
    res.redirect(oldUrl);
  } else {
    res.redirect('/user/profile');
  }
});
```

Ngoài ra chúng ta còn có thể thiết lập đang nhập bằng tài khoản Facobook, Google, Twitter,...

VD: Đăng nhập với tài khoản Facebook:

1. Cấu hình.
   Tạo ứng dụng tại Facebook Developers, sau khi tạo một ứng dụng được chỉ định một App ID và App Secret. Ứng dụng của bạn cũng phải thực hiện một URL chuyển hướng, mà Facebook sẽ chuyển hướng người dùng sau khi họ đã chấp thuận quyền truy cập vào ứng dụng của bạn.

```
var passport = require('passport')
  , FacebookStrategy = require('passport-facebook').Strategy;


passport.use(new FacebookStrategy({
    clientID: FACEBOOK_APP_ID,
    clientSecret: FACEBOOK_APP_SECRET,
    callbackURL: "http://www.example.com/auth/facebook/callback"
  },
  function(accessToken, refreshToken, profile, done) {
    User.findOrCreate(..., function(err, user) {
      if (err) { return done(err); }
      done(null, user);
    });
  }
));
```

2. Routes

Chuyển hướng người dùng đăng nhập

```
// Redirect the user to Facebook for authentication.  When complete,
// Facebook will redirect the user back to the application at
//     /auth/facebook/callback
app.get('/auth/facebook', passport.authenticate('facebook'));

// Facebook will redirect the user to this URL after approval.  Finish the
// authentication process by attempting to obtain an access token.  If
// access was granted, the user will be logged in.  Otherwise,
// authentication has failed.
app.get('/auth/facebook/callback',
  passport.authenticate('facebook', { successRedirect: '/',
                                      failureRedirect: '/login' }));
```

3. Cho phép truy cập.

```
app.get('/auth/facebook',
  passport.authenticate('facebook', { scope: 'read_stream' })
);
```

Or

```
app.get('/auth/facebook',
  passport.authenticate('facebook', { scope: ['read_stream', 'publish_actions'] })
);
```

3.5: Phiên làm việc.

Đối với việc thanh toán mua hàng qua mạng việc bảo mật thông tin tài khoản mật khẩu vô cùng quan trọng. Ví dụ đối với các website ngân hàng việc lưu trữ tài khoản, mật khẩu với thời gian phiên làm việc của khách hàng thường sẽ là 3 phút nếu khách hàng không có bất kì hành động thao tác nào với trang web thì phiên làm việc của khách hàng sẽ hết nếu muốn quay lại thì khách hàng cần phải đăng nhập lại tài khoản, mật khẩu.

```
app.use(session({
  secret: 'mysupersecret',
  resave: true,
  saveUninitialized: true,
  store: new MongoStore({ mongooseConnection: mongoose.connection }),
```

```
  cookie: { maxAge: 30 * 60 * 1000 }
}));
```

Ở đây tài khoản mật khẩu của khách hàng sẽ được lưu vào cookie trong vòng 3 phút.

3.5: Thanh toán trực tuyến thông qua các công ty, nhà cung cấp các dịch vị thương mại điện tử.

Việc thanh toán truyến ngày càng phổ biến chỉ với một chiếc thẻ tín dụng ngân hàng trên thế giới có các công ty chuyên cung cấp dịch vụ thanh toans trực tuyến như Paypal, Stripe,...

Trong npm có module stripe.js giúp chúng ta xây dựng được chức năng này một cách dễ dàng.

1.Để sử dụng ta cần include thư viện stripe.js

```
<script src="https://js.stripe.com/v3/"></script>
```

2.Tạo đối tượng charges.

```
var stripe = require("stripe")(
  "sk_test_FSiDCU0Hbbml5K2VQFhAde8c"
);

stripe.charges.create({
  amount: 2000,
  currency: "usd",
  source: "tok_1AMfaXBkQ1C3wsaYjrCnbQJi", // obtained with Stripe.js
  description: "Charge for michael.jones@example.com"
}, function(err, charge) {
  // asynchronously called
});
```

3.Tìm kiếm.

```
var stripe = require("stripe")(
  "sk_test_FSiDCU0Hbbml5K2VQFhAde8c"
);

stripe.charges.retrieve(
  "ch_1ATOPWBkQ1C3wsaYWm7cvT4O",
```

```
  function(err, charge) {
    // asynchronously called
  }
);
```

4.Cập nhật.

```
var stripe = require("stripe")(
  "sk_test_FSiDCU0Hbbml5K2VQFhAde8c"
);

stripe.charges.update(
  "ch_1ATOPWBkQ1C3wsaYWm7cvT4O",
  {
    description: "Charge for michael.jones@example.com"
  },
  function(err, charge) {
    // asynchronously called
  }
);
```

5.Thu nạp

```
var stripe = require("stripe")(
  "sk_test_FSiDCU0Hbbml5K2VQFhAde8c"
);

stripe.charges.capture("ch_1ATOPWBkQ1C3wsaYWm7cvT4O", function(err, charge) {
  // asynchronously called
});
```

VD: Tài khoản thanh toán thành công.

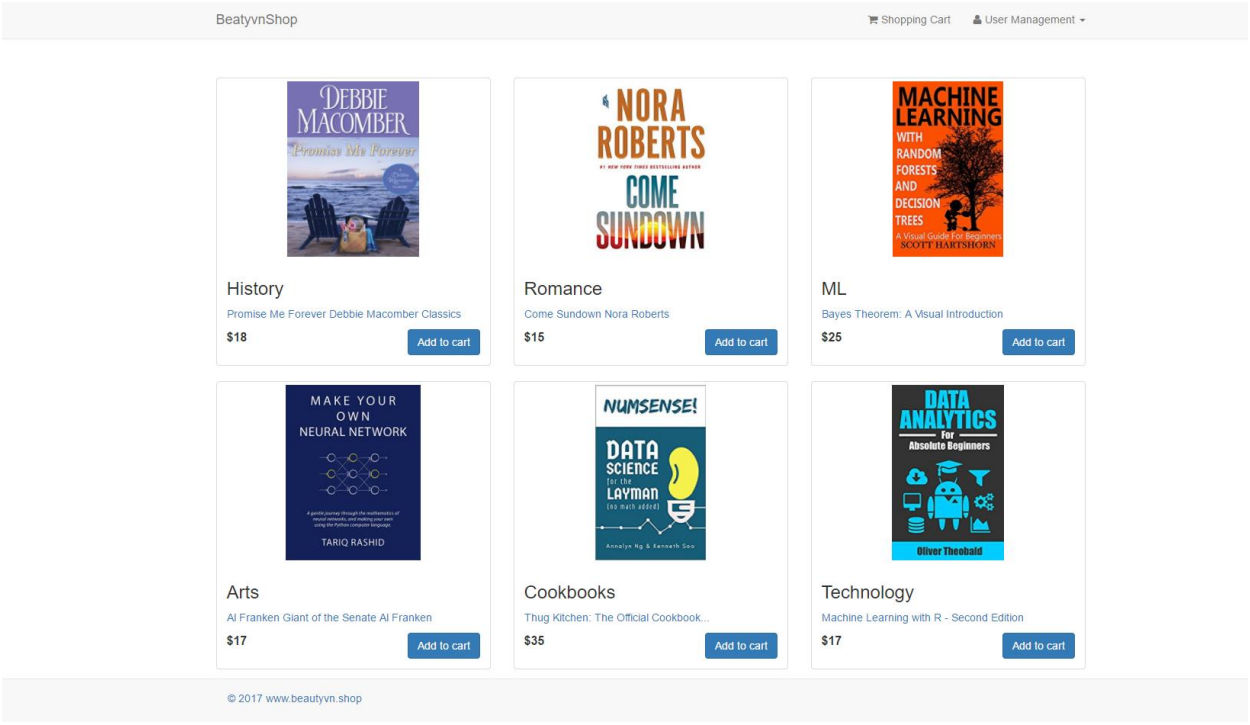

```
{
  "object": {
    "id": "ch_1ATOPWBkQ1C3wsaYWm7cvT4O",
    "object": "charge",
    "amount": 2500,
```

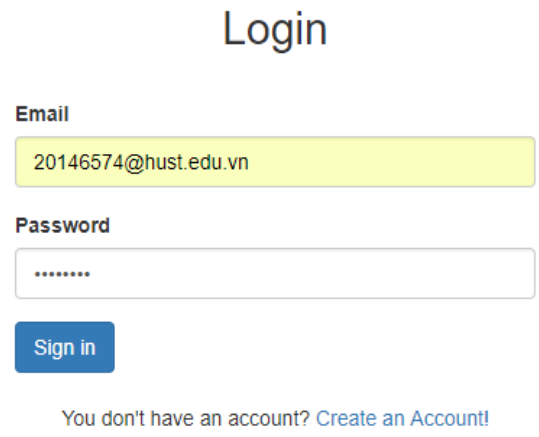
```
"amount_refunded": 0,
"application": null,
"application_fee": null,
"balance_transaction": "txn_1ATOPWBkQ1C3wsaY6j8sfXwc",
"captured": true,
"created": 1497158090,
"currency": "usd",
"customer": null,
"description": "Customer charges for Beautyvn-Shop",
"destination": null,
"dispute": null,
"failure_code": null,
"failure_message": null,
"fraud_details": {
},
"invoice": null,
"livemode": false,
"metadata": {
},
"on_behalf_of": null,
"order": null,
"outcome": {
  "network_status": "approved_by_network",
  "reason": null,
  "risk_level": "normal",
  "seller_message": "Payment complete.",
  "type": "authorized"
},
"paid": true,
"receipt_email": null,
"receipt_number": null,
"refunded": false,
"refunds": {
  "object": "list",
  "data": [
  ],
  "has_more": false,
```

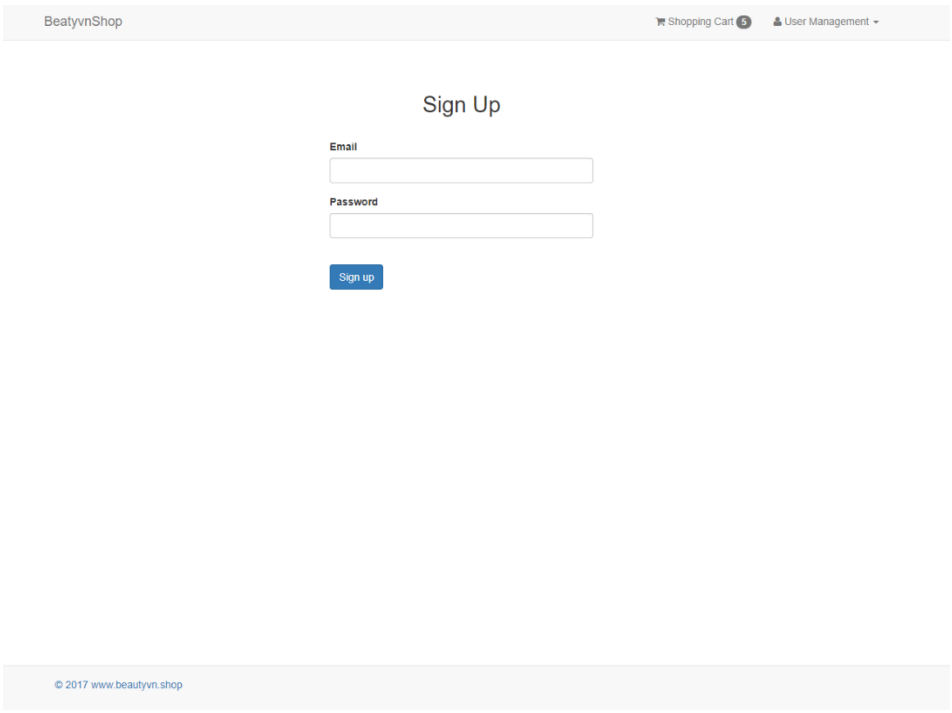
```
      "total_count": 0,
      "url": "/v1/charges/ch_1ATOPWBkQ1C3wsaYWm7cvT4O/refunds"
    },
    "review": null,
    "shipping": null,
    "source": {
      "id": "card_1ATOPVBkQ1C3wsaYEuiDNIBg",
      "object": "card",
      "address_city": null,
      "address_country": null,
      "address_line1": null,
      "address_line1_check": null,
      "address_line2": null,
      "address_state": null,
      "address_zip": null,
      "address_zip_check": null,
      "brand": "MasterCard",
      "country": "US",
      "customer": null,
      "cvc_check": "pass",
      "dynamic_last4": null,
      "exp_month": 4,
      "exp_year": 2021,
      "fingerprint": "cMxPQkbmBzcfCiY0",
      "funding": "credit",
      "last4": "4444",
      "metadata": {
      },
      "name": "Nguyen Dinh Quang",
      "tokenization_method": null
    },
    "source_transfer": null,
    "statement_descriptor": null,
    "status": "succeeded",
    "transfer_group": null
  }
}
```

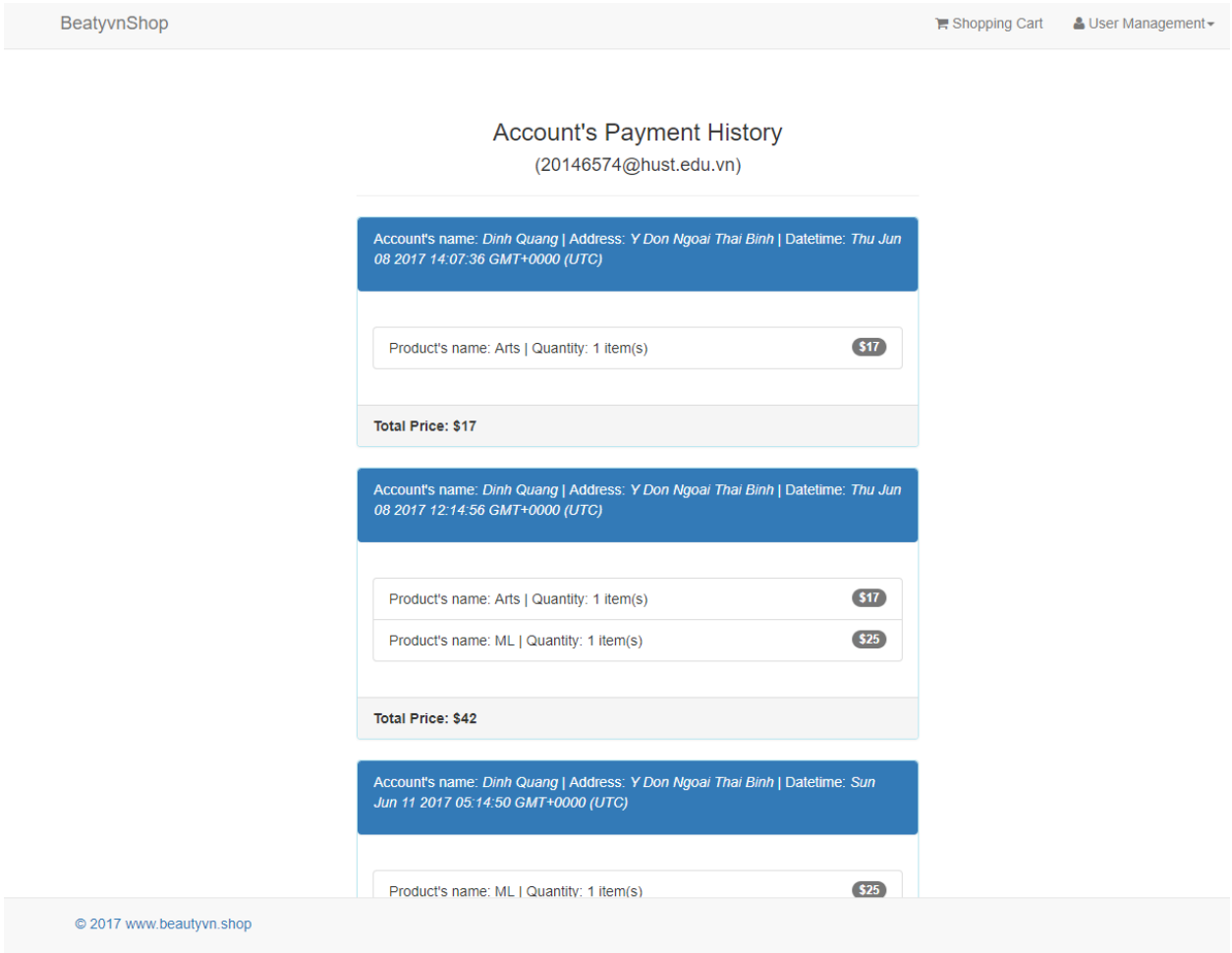
3.6: Một số hình ảnh minh họa.

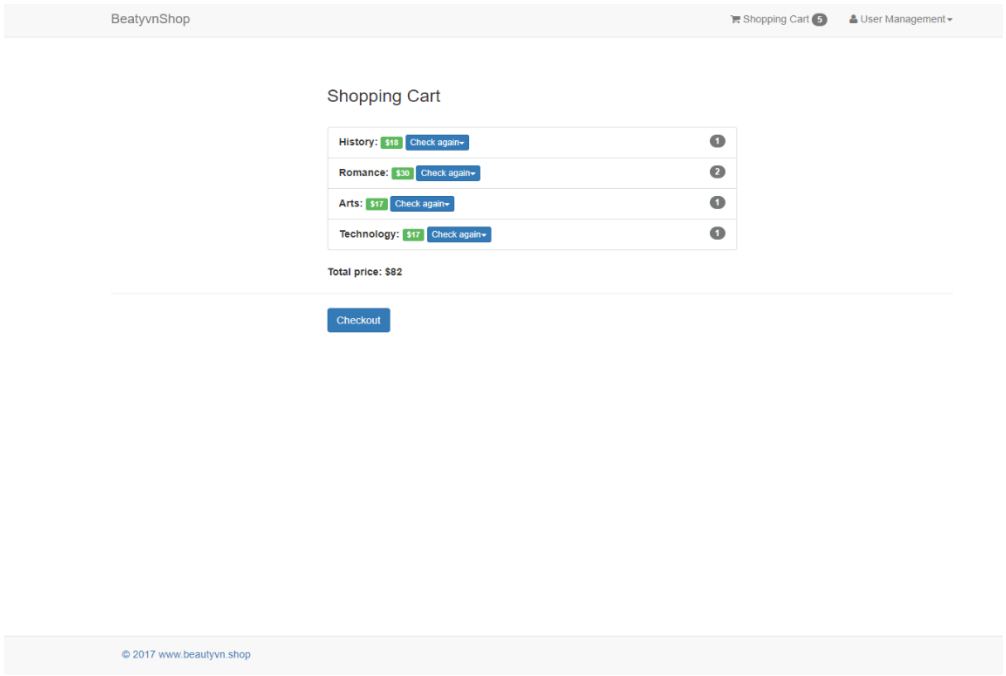
3.6.1: Giao diện website

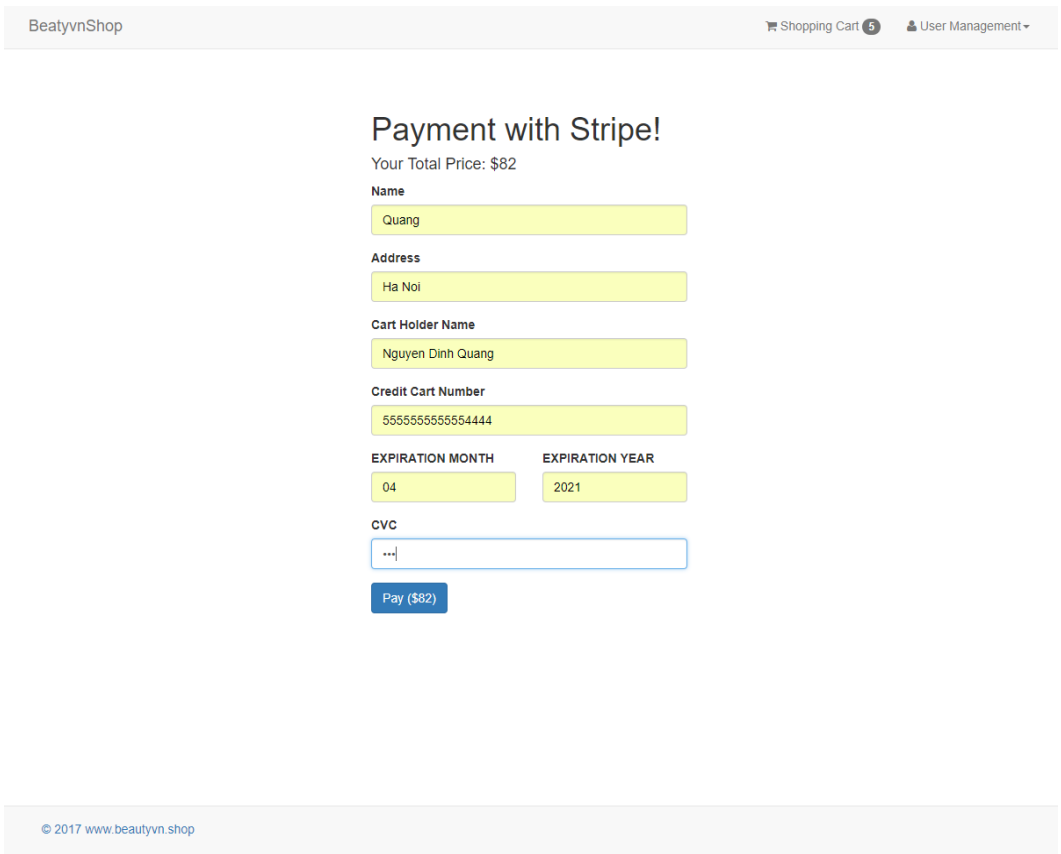





BeatyvnShop
Shopping Cart 5
User Management
Sign Up
Email
Password
Sign up
© 2017 www.beautyvn.shop

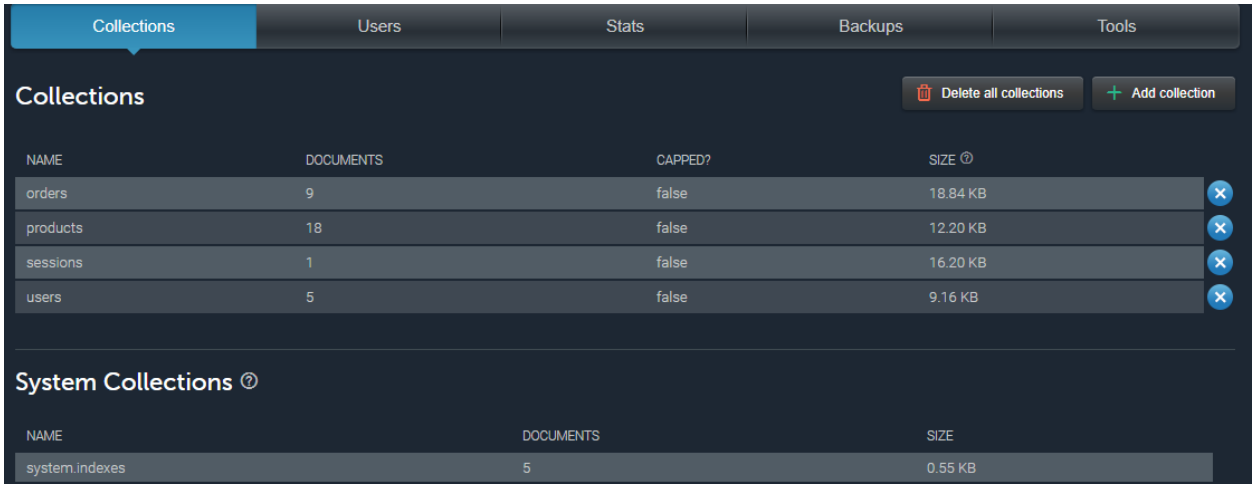

BeatyvnShop
Shopping Cart
User Management
Account's Payment History
(20146574@hust.edu.vn)
Account's name: Dinh Quang | Address: Y Don Ngoai Thai Binh | Datetime: Thu Jun 08 2017 14:07:36 GMT+0000 (UTC)
Product's name: Arts | Quantity: 1 item(s)
$17
Total Price: $17
Account's name: Dinh Quang | Address: Y Don Ngoai Thai Binh | Datetime: Thu Jun 08 2017 12:14:56 GMT+0000 (UTC)
Product's name: Arts | Quantity: 1 item(s)
$17
Product's name: ML | Quantity: 1 item(s)
$25
Total Price: $42
Account's name: Dinh Quang | Address: Y Don Ngoai Thai Binh | Datetime: Sun Jun 11 2017 05:14:50 GMT+0000 (UTC)
Product's name: ML | Quantity: 1 item(s)
$25
© 2017 www.beautyvn.shop


BeatyvnShop
Shopping Cart 5
User Management
Shopping Cart
History: $18 Check again 1
Romance: $30 Check again 2
Arts: $17 Check again 1
Technology: $17 Check again 1
Total price: $82
Checkout
© 2017 www.beautyvn.shop


BeatyvnShop
Shopping Cart 5
User Management
Payment with Stripe!
Your Total Price: $82
Name
Quang
Address
Ha Noi
Cart Holder Name
Nguyen Dinh Quang
Credit Cart Number
5555555555554444
EXPIRATION MONTH
04
EXPIRATION YEAR
2021
CVC
Pay ($82)
© 2017 www.beautyvn.shop

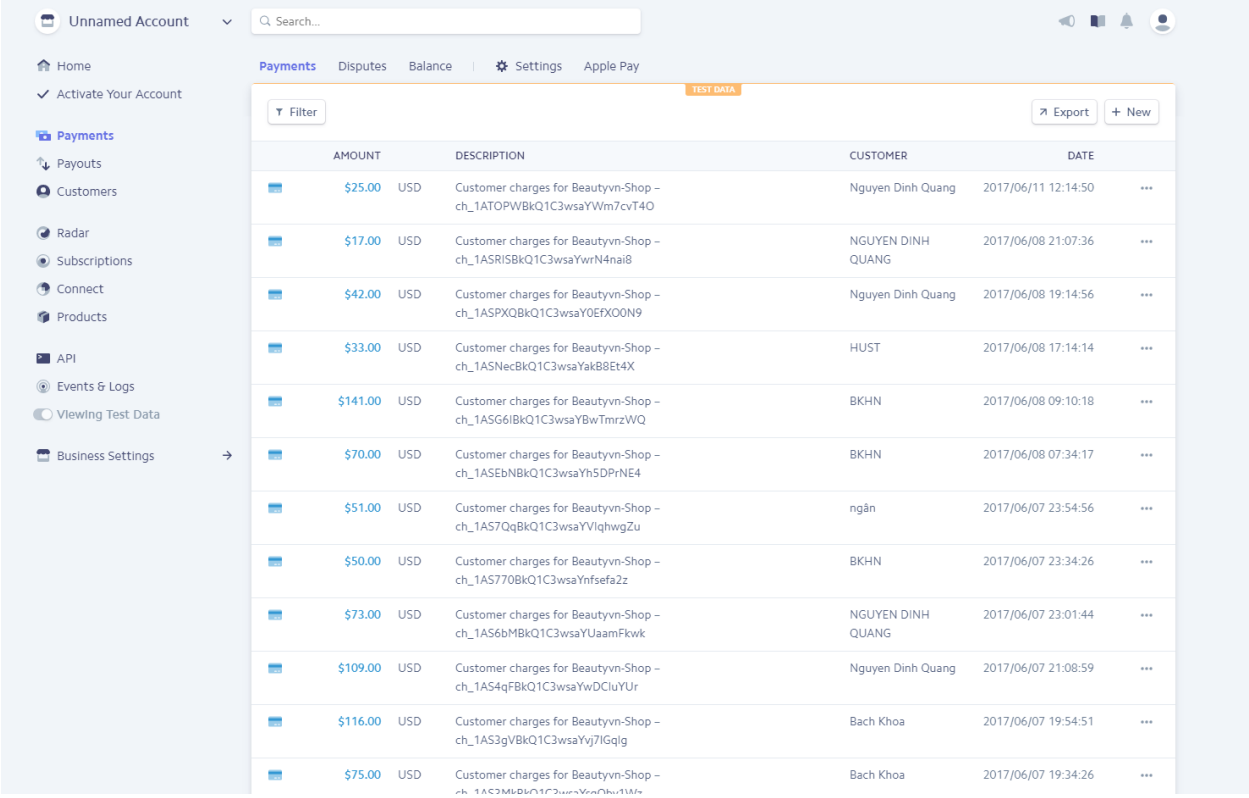
3.6.2:Database

3.6.2.1: Trên mlab.com



3.6.2.2: Database trên localhost

3.6.3:Thanh toán với Stripe



Phần III: Hướng phát triển.

+ Phân trang
+Page details sản phẩm
+Đăng nhập bằng tài khoản Facebook, Google, ...
+Xác thực tài khoản bằng Email,
+Contact Form Emai, ...
+Giao diện cần được quan tâm, đầu tư thêm.

Tài liệu tham khảo

Nodejs.org

Express

NPM

Youtube.com, ...