

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN



KIẾN TRÚC HƯỚNG DỊCH VỤ

ĐỒ ÁN CUỐI KÌ

Microservices – Based Ecommerce Website

Người hướng dẫn: Thầy **DƯƠNG HỮU PHÚC**

Người thực hiện: **CHIÊM TIỀN KHANG – 518H0516**

NGUYỄN PHẠM MẠNH CƯỜNG – 518H0479

NGUYỄN CHÂU THÙY LINH – 518H0031

Lớp : **18H50203**

18H50202

Khoá : **22**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022

**TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN**



KIẾN TRÚC HƯỚNG DỊCH VỤ

ĐỒ ÁN CUỐI KÌ

**Microservices – Based Ecommerce
Website**

Người hướng dẫn: Thầy **DƯƠNG HỮU PHÚC**

Người thực hiện: **CHIÊM TIỀN KHANG – 518H0516**

NGUYỄN PHẠM MẠNH CƯỜNG – 518H0479

NGUYỄN CHÂU THÙY LINH – 518H0031

Lớp : **18H50203**

18H50202

Khoá : **22**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 202

LỜI CẢM ƠN

Xin chân thành cảm ơn Khoa CNTT, Trường Đại học Tôn Đức Thắng đã tạo điều kiện thuận lợi cho chúng tôi nghiên cứu và thực hiện chuyên đề này.

Chân thành cảm ơn thầy Dương Hưu Phúc đã hỗ trợ và giảng dạy cho chúng em trong suốt quá trình học tập để chúng em có thể thực hiện bài báo cáo này.

Mặc dù em đã cố gắng hoàn thành luận văn trong phạm vi và khả năng nhưng chắc chắn sẽ có những thiếu sót. Rất mong nhận được sự thông cảm, góp ý, chỉ bảo của thầy và các bạn

ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng em xin cam đoan đây là sản phẩm đồ án của riêng chúng em và được sự hướng dẫn của thầy Dương Hữu Phúc. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào chúng em xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do chúng em gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 21 tháng 05 năm 2022

Tác giả

(ký tên và ghi rõ họ tên)

*Chiêm Tiên Khang
Nguyễn Phạm Mạnh Cường
Nguyễn Châu Thùy Linh*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

1. Phần xác nhận của GV hướng dẫn

Tp. Hồ Chí Minh, ngày tháng
năm (kí và ghi họ tên)

2. Phần đánh giá của GV chấm bài

Tp. Hồ Chí Minh, ngày tháng năm (kí và ghi họ tên)

Table of Contents

| | |
|--|----|
| LỜI CẢM ƠN | 3 |
| PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN..... | 5 |
| 1. Phần xác nhận của GV hướng dẫn | 5 |
| 2. Phần đánh giá của GV chấm bài | 5 |
| TECH STACK | 7 |
| 1 NodeJS | 7 |
| 1.1 What is Node.js ? | 7 |
| 1.2 Features of Node.js | 7 |
| 1.3 Who and When to use Node.js | 7 |
| 1.4 Concepts | 8 |
| 2. ExpressJS | 8 |
| 2.1 What is ExpressJS ? | 8 |
| 2.2 Why Express ? | 8 |
| 2.3 MongoDB and Mongoose | 8 |
| 2.4 MongoDB Overview | 9 |
| 2.4.1 What is MongoDB ? | 9 |
| 2.4.2 Database | 9 |
| 2.4.3 Collection | 9 |
| 2.4.4 Document | 9 |
| APPLICATION SYSTEM DESIGN..... | 10 |
| 1. Web App Architecture | 10 |
| 1.1 Microservices | 10 |
| 1.2 The Node.js Platform | 11 |
| 1.3 Building Microservices with Node.js | 12 |
| 2. Functional requirement and Non-functional requirement | 12 |
| 3. API Documentations | 13 |
| 3. DATABASE DESIGN | 14 |
| 4. FRONT-END DESIGN..... | 16 |
| 5 REFERENCE | 20 |

TECH STACK

1 NodeJS

1.1 What is Node.js ?

- Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009 and its latest version is v0.10.36.
- Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

1.2 Features of Node.js

- **Asynchronous and Event Driven** – All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.
- **Very Fast** – Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.
- **Single Threaded but Highly Scalable** – Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.
- **No Buffering** – Node.js applications never buffer any data. These applications simply output the data in chunks.

1.3 Who and When to use Node.js

- Following is the link on github wiki containing an exhaustive list of projects, application and companies which are using Node.js. This list includes eBay, General Electric, GoDaddy, Microsoft, PayPal, Uber, Wikipins, Yahoo!, and Yammer to name a few
- **Where to use Node.js:**
 - I/O bound Applications
 - Data Streaming Applications
 - Data Intensive Real-time Applications (DIRT)

- JSON APIs based Applications
- Single Page Applications

1.4 Concepts

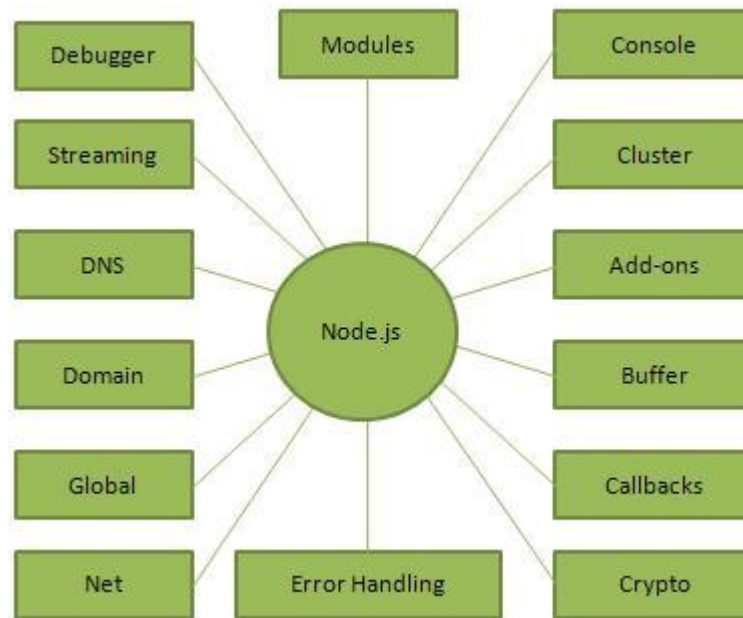


Figure 1 NodeJS

2. ExpressJS

2.1 What is ExpressJS ?

- ExpressJS is a web application framework that provides you with a simple API to build websites, web apps and back ends. With ExpressJS, you need not worry about low level protocols, processes, etc.
- Express provides a minimal interface to build our applications. It provides us the tools that are required to build our app. It is flexible as there are numerous modules available on npm, which can be directly plugged into Express.
- Express was developed by TJ Holowaychuk and is maintained by the Node.js foundation and numerous open source contributors.

2.2 Why Express ?

- Unlike its competitors like Rails and Django, which have an opinionated way of building applications, Express has no "best way" to do something. It is very flexible and pluggable.

2.3 MongoDB and Mongoose

- MongoDB is an open-source, document database designed for ease of

development and scaling. This database is also used to store data.

- Mongoose is a client API for node.js which makes it easy to access our database from our Express application

2.4 MongoDB Overview

2.4.1 What is MongoDB ?

- MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document.

2.4.2 Database

- Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

2.4.3 Collection

- Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

2.4.4 Document

A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

| RDBMS | MongoDB |
|----------------------------|--|
| Database | Database |
| Table | Collection |
| Tuple/Row | Document |
| column | Field |
| Table Join | Embedded Documents |
| Primary Key | Primary Key (Default key _id provided by MongoDB itself) |
| Database Server and Client | |
| mysqld/Oracle | mongod |
| mysql/sqlplus | mongo |

Figure 2 The relationship of RDBMS terminology with MongoDB.

APPLICATION SYSTEM DESIGN

1. Web App Architecture

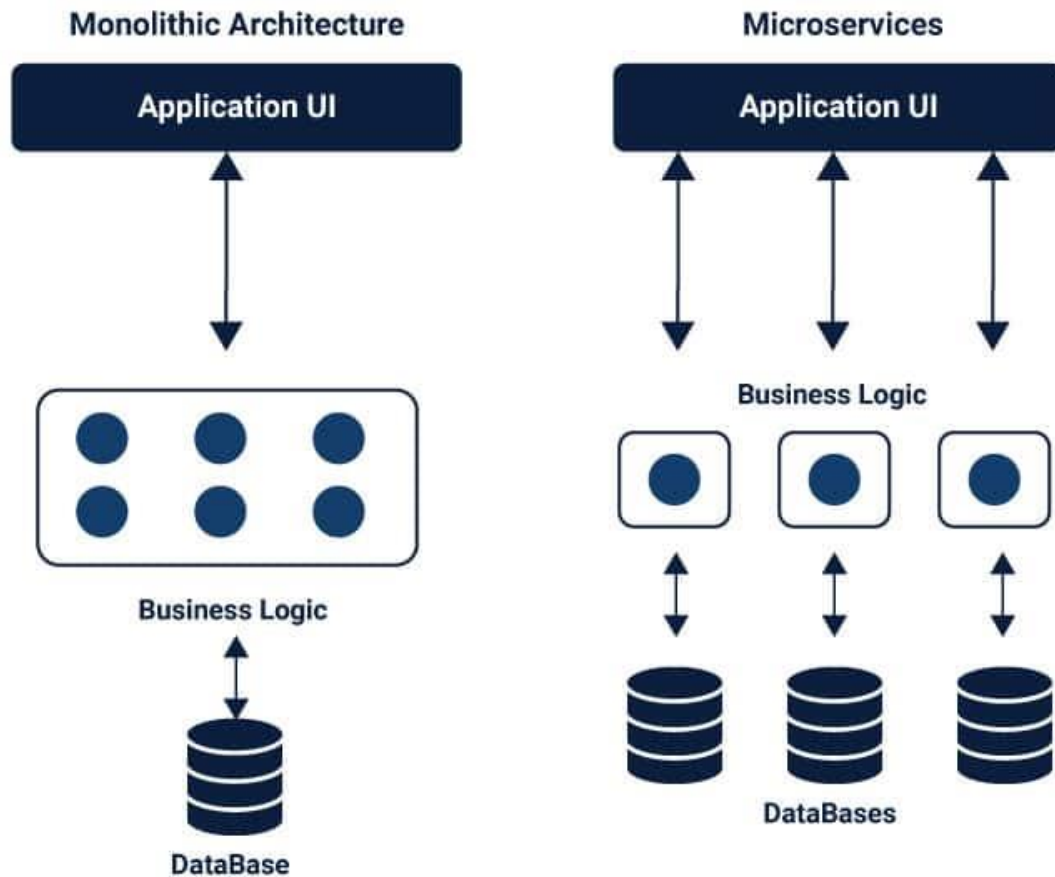
1.1 Microservices

In software application development, microservices are a style of service-oriented architecture (SOA) where the app is structured on an assembly of interconnected services. With microservices, the application architecture is built with lightweight protocols. The services are finely seeded in the architecture. Microservices disintegrate the app into smaller services and enable improved modularity.

Compared to its predecessor, the monolithic architecture, microservices are hands down more beneficial. You don't need to stuff all software components and services in one big container and pack them tightly. With microservices, you can build an app with:

- greater flexibility,
- high scalability,
- continuous development,
- systematic data organization,
- time optimization, and
- reliability.

Building JavaScript applications on microservices help you focus on developing monofunctional modules with clearly defined operations and precise interfaces. The application development process becomes more agile, and the challenges of continuous testing are mitigated.



1.2 The Node.js Platform

When you build applications on a monolithic architecture, the entire application needs to be deployed with every update. On the other hand, microservices have no dependency on the type of framework, technique or programming language being used to build them. Your ability to release REST-ful APIs for communication and other services is the only requisite for microservice architecture.

Key benefits of Node.js

- **Single-threaded:** With event looping, the server uses a non-blocking mechanism to respond.
- **Super-fast:** Codes are executed quickly on the V8 JavaScript Engine
- **Event-driven:** 'Events of Node.js' is a notification system that enables the application server to capture the response of the previous API call.
- **Buffer-less:** There is no buffering as data is simply released in chunks.
- **Asynchronous:** The non-blocking, non-synchronous Node.js libraries

move to the next API and do not await the return data of the previous API.

- Highly-scalable: Servers can handle as many requests as coming their way
- Licensed: The program is authorized under a software license.

1.3 Building Microservices with Node.js

To build microservices for real-world applications with Node.js, a basic understanding of JavaScript programming is important. The steps involved in developing microservices with Node.js showcase how working applications in our hyperconnected world can perform superbly when constructed with a functional combination of multiple, unique APIs.

2. Functional requirement and Non-functional requirement

- Login:

- Functional requirement: Correct email and password.
- Non-functional requirement: send OTP or email code to confirm for first login

- Register:

- Functional requirement: email, password, fullname, address, phone, age are required
- Non-functional requirement: email must be unique

- Logout:

- Functional requirement: must be logged in before
- Non-functional requirement: Log out all devices (optional)

- Get product list (all products):

- Functional requirement: Connect to database to get product data
- Non-functional requirement: Full product information available

- Get product details:

- Functional requirement: full information of each product
- Non-functional requirement: format number of as currency

- Add product:

- Functional requirement: full information of each product
- Non-functional requirement: correct format of that product

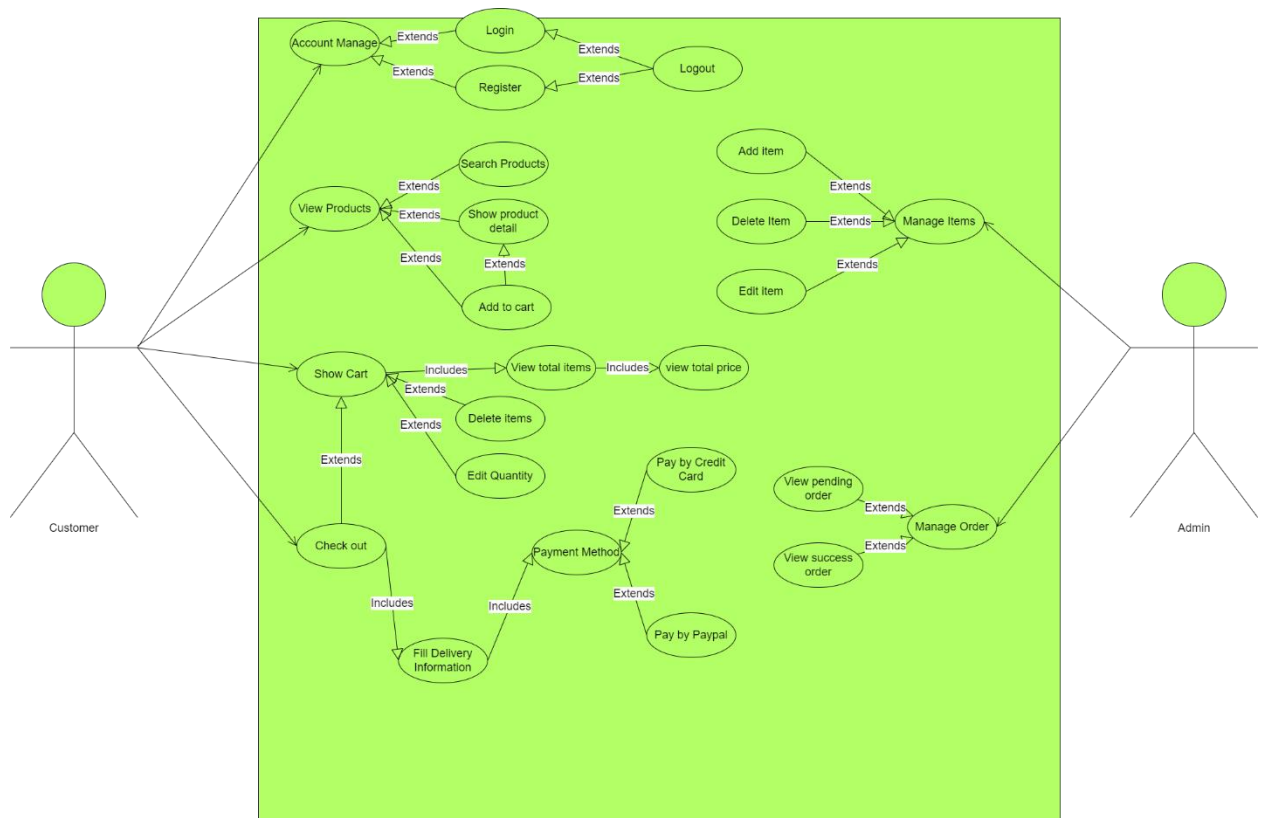
3. API Documentations

| End point (URI) | HTTP Method | Input | Output | Description |
|-----------------------|-------------|--|-----------------------------|-----------------------------|
| /register | POST | email, password, fullname, address, phone, age | user, token | Create new user |
| /login | POST | email, password | user, token | User sign in into system |
| /users/me | GET | user's token | user infomation | Get user information |
| /users/me/logout | POST | token | message: "Logged out" | User sign out from system |
| /users/me/logoutAll | POST | token | message: "Logged out" | Log out all devices |
| /api/products | GET | | products | Get all product information |
| /api/products/details | GET | | product | Get details of a product |

| | | | | |
|--------------------------|--------|--|----------------------------------|----------------------------|
| /api/products | POST | name, brand, description, price, image, category, quantity, stock | product | Create products |
| /api/products | DELETE | productID | message: "Deleted" | Delete a product |
| /api/products/:productId | GET | productID | product | Get product ID |
| /api/products/:productId | PUT | productID | product | Update product information |
| /api/cart | GET | products, totalPrice | products, totalPrice, cart | Get cart information |
| /api/cart | POST | products | cart | Add products into cart |
| /api/cart | DELETE | productID | cart | Delete a product |
| /api/order | POST | user, address, phone, cart | result | Create order |

3. DATABASE DESIGN

Use-Case:



4. FRONT-END DESIGN

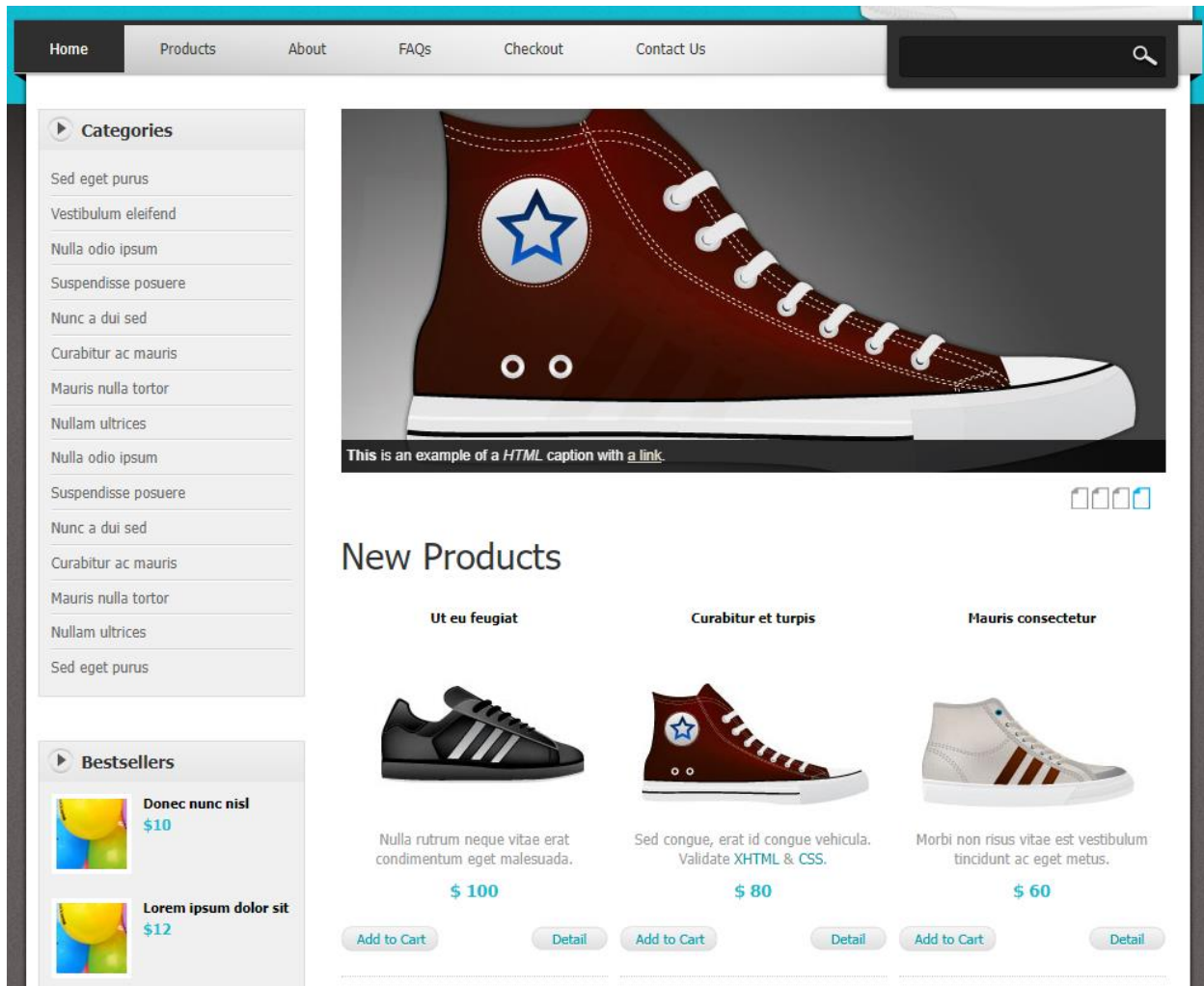


Figure 3 Homepage

Products







| | | |
|---|---|--|
| <p>Ut eu feugiat</p>  <p>Fusce in dui et neque malesuada tincidunt nec at urna. Validate XHTML & CSS.</p> <p>\$ 100</p> <div><a>Add to Cart<a>Detail</div> | <p>Curabitur et turpis</p>  <p>Etiam et sapien ut nunc blandit euismod. Sed dui libero, semper a volutpat sed, placerat eu lectus.</p> <p>\$ 80</p> <div><a>Add to Cart<a>Detail</div> | <p>Mauris consectetur</p>  <p>Curabitur pellentesque ullamcorper massa ac ultricies. Maecenas porttitor erat quis leo pellentesque.</p> <p>\$ 60</p> <div><a>Add to Cart<a>Detail</div> |
| <p>Proin volutpat</p>  <p>Morbi non risus vitae est vestibulum tincidunt ac eget metus. Sed congue, erat id congue vehicula.</p> <p>\$ 260</p> <div><a>Add to Cart<a>Detail</div> | <p>Aenean tempus</p>  <p>Aenean eu elit arcu. Quisque eget blandit erat. Integer molestie malesuada augue vitae mollis.</p> <p>\$ 80</p> <div><a>Add to Cart<a>Detail</div> | <p>Nulla luctus urna</p>  <p>Nunc nisl nisi, aliquet eu gravida vitae, porta vel ante. Pellentesque faucibus risus et sem volutpat.</p> <p>\$ 190</p> <div><a>Add to Cart<a>Detail</div> |

Figure 4 Products Page

Product Detail



Price: \$100
Availability: In Stock
Model: Product 14
Manufacturer: Apple
Quantity:

[Add to Cart](#)

Product Description

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur semper quam sit amet turpis rhoncus id venenatis tellus sollicitudin. Fusce ullamcorper, dolor non mollis pulvinar, turpis tortor commodo nisl, et semper lectus augue blandit tellus. Quisque id bibendum libero. Validate [XHTML](#) & [CSS](#).

Related Products



Ut eu feugiat

\$ 100

[Add to Cart](#)



Curabitur et turpis

\$ 200

[Detail](#)

[Add to Cart](#)



Mauris consectetur

\$ 120

[Add to Cart](#)

[Detail](#)

Figure 5 Product's Detail

Checkout

BILLING INFORMATION

Full Name (must be same as on your credit card):

Address:

City:

Country:

E-MAIL

PHONE

Please, specify your reachable phone number. YOU MAY BE GIVEN A CALL TO VERIFY AND COMPLETE THE ORDER.

SHOPPING CART

TOTAL AMOUNT: **\$240**

☐ I accept the [terms of use](#) of this website.



Recommended if you have a PayPal account. Fastest delivery time.

[PAYPAL](#)



2Checkout.com, Inc. is an authorized retailer of goods and services. 2CheckOut accepts customer orders via online checks, Visa, MasterCard, Discover, American Express, Diners, JCB and debit cards with the Visa, Mastercard logo. Validate [XHTML](#) & [CSS](#).

[2CHECKOUT](#)

Figure 6 Checkout

Shopping Cart




| Image | Description | Quantity | Price | Total | |
|--|--|--------------------------------|--------------|--------------|------------------------|
|  | Etiam in tellus (Validate XHTML & CSS) | <input type="text" value="1"/> | \$100 | \$100 | Remove |
|  | Second Red Shoes | <input type="text" value="1"/> | \$80 | \$80 | Remove |
|  | Hendrerit justo | <input type="text" value="1"/> | \$60 | \$60 | Remove |
| Have you modified your basket? Please click here to Update | | | Total | \$240 | |
| Proceed to checkout | | | | | |
| Continue shopping | | | | | |

Figure 7 View Cart

5 REFERENCE

[Node.js - Introduction \(tutorialspoint.com\)](#)

[MongoDB - Overview \(tutorialspoint.com\)](#)

[Node.js - Express Framework \(tutorialspoint.com\)](#)