

# REPORT

On

# Professional Practice in It Project

*Submitted*

*In partial fulfillment*

*For the award of the Degree of*

*Bachelor of Science*

*in Computing in Software Development (year 3)*



**Submitted By:**

**Muhammad Luqman (G00353385)**

**Muhammad Noman Junaid (G00351754)**

**GitHub Repository Link = <https://github.com/LuqmanFarooq/-Professional-Practice-in-IT>**

**Department of Computer Science**

## *Candidate's Declaration*

We hereby declare that the work, which is being presented in the Project Report, entitled “Shopping Hunt” in fulfillment for the module Professional Practice in it, is a record of our investigations carried under the Guidance of Dr.Martin Kenirons Department of Computer Science.

## Table of Contents

Professional Practice in It Project.....	1
<i>Candidate's Declaration</i> .....	2
Introduction.....	6
PROJECT IDEA .....	6
OBJECTIVES.....	6
Project Description.....	6
Client Area .....	7
Homepage.....	7
About Us .....	7
Contact Us.....	8
All Products.....	9
Category .....	9
Product Detail page .....	10
My Cart.....	11
Register.....	13
Login.....	14
Admin Area .....	15
Admin Pages .....	15
Admin Categories .....	16
Admin Product .....	17
System Requirements .....	19
Hardware Requirements.....	19
Software Requirements .....	19
System Architecture .....	20
UI.....	20
Scripting.....	20
Server-Side Scripting .....	20

Client-Side Scripting .....	20
DataBase.....	20
Data Flow Diagrams .....	21
Use case Diagram .....	23
Technology Used and Why .....	23
Node .....	24
How Does It Work? .....	24
Why Node.js .....	26
Express .....	26
Why Express .....	27
EJS .....	28
Why EJS? .....	28
MongoDB.....	28
Why MongoDB .....	28
Mongoose .....	29
Why Mongoose .....	30
AJAX .....	30
What Is AJAX?.....	30
How Ajax Works?.....	31
Why Ajax.....	31
Session Storage .....	32
Why use Session Storage? .....	32
Authentication .....	33
Why salting of password is needed.....	34
Bootstrap .....	34
Why Bootstrap .....	34
HTML .....	36
Why HTML.....	37

CSS.....	39
Why CSS .....	40
Limitations and Known Bugs .....	40
Testing Plans .....	40
Recommendations for Future Development .....	40
Conclusions .....	40
References.....	41

# Introduction

## PROJECT IDEA

For this project, we had to come up with something related to our Course.

We browsed the internet for ideas and then in the first meeting with our project supervisor Dr. Martin Kenirons and decided to design and develop an E-commerce Website.

Muhammad Noman Junaid did the most front-end development and Muhammad Luqman did the Backend. And obviously, as it's a group project so for most parts of the project we worked together to decide technologies we should use and how to implement them.

## OBJECTIVES

Online Shopping is the process whereby consumers directly buy goods and services without any intermediary service over the internet. The goal of this website is to develop a web-based interface for sellers to sell their goods and buyers to buy without going out to the shop their selves, the website would be easy to use and hence the shopping experience pleasant for the users. The main goal of this website is:

1. To develop an easy to use web-based interface where user can browse products in categories, view a complete description of the product and order the product.
2. A user can buy products from home and get delivery.
3. An admin/seller can sell products from home.

## Project Description

Our project is Shopping Hunt. This is a Web Application which helps clients to buy the products online and seller to list his products for Selling. Shopping Hunt is an interactive e-commerce solution providing users with an opportunity to buy and sell Products and supports PayPal as well for buying.

In this Web Application, we have basically 2 modules. The first module includes the customer module and the second module includes an admin module.

The customer must register for Buying Products. The registered customer can view details of products and he/she can buy products of his/her need. He/she has to pay and will get home delivery.

The admin module contains the access to the admin page on the website. The admin can change everything on the website. He can add, delete, and update pages, categories, products, and any other information of the web Applications.

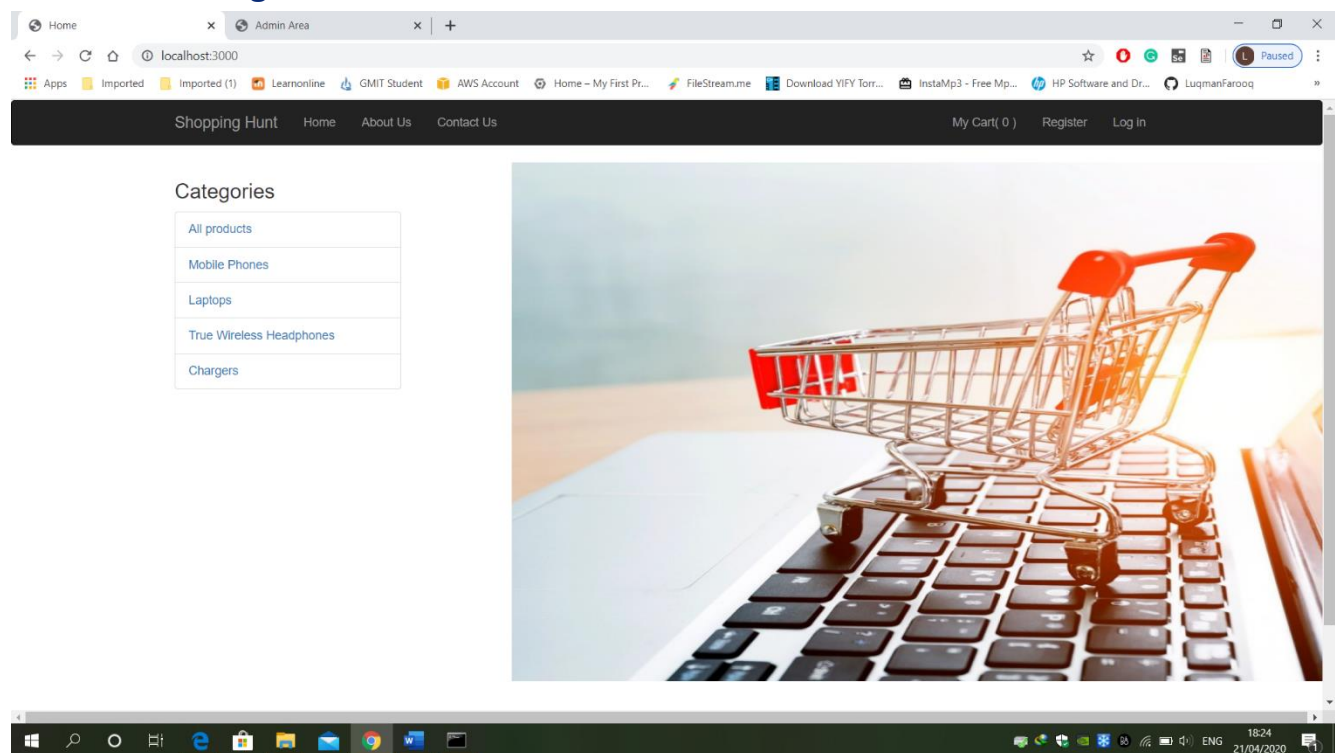
## Client Area

### Homepage

The home page will show an intro page introducing the user to the webApp.

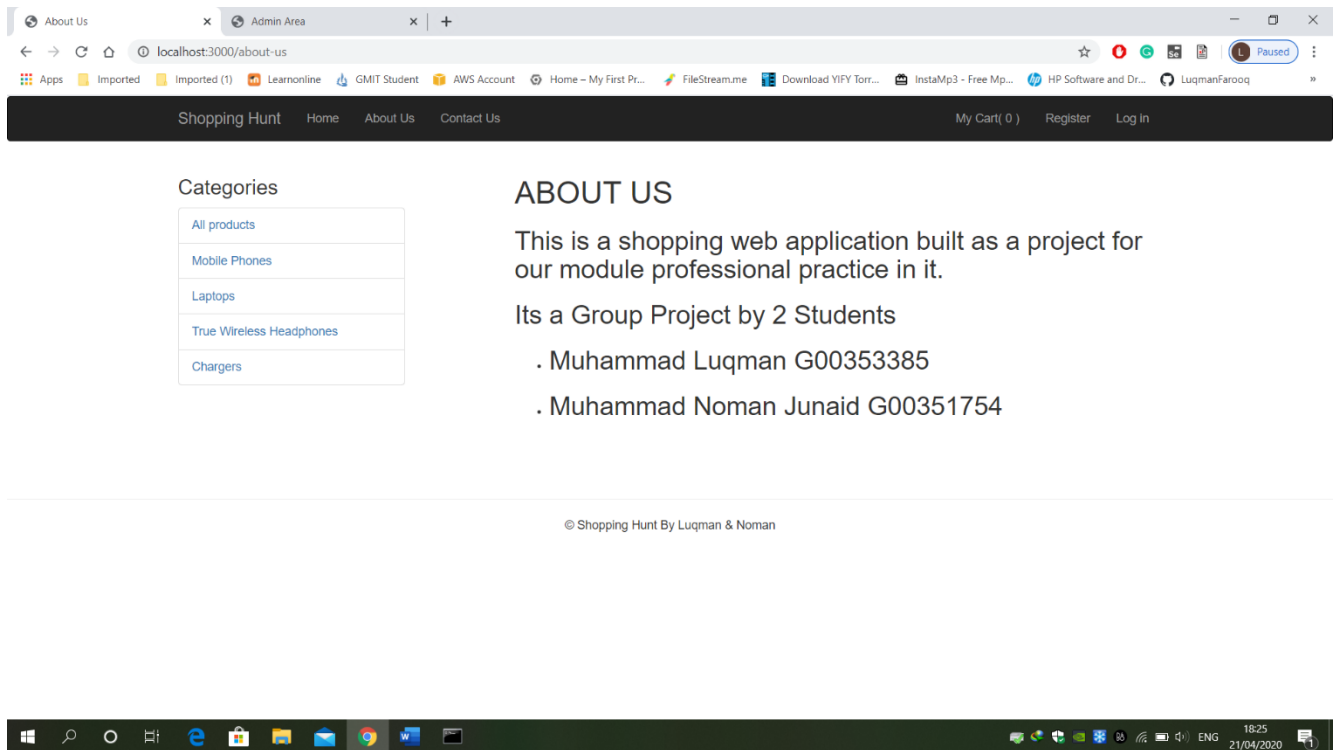
It has a clickable title which navigates to the home page. A navigation bar shown on all pages with links to Home, About Us, Contact Us, My Cart with a count of items in the cart if any, Register and log in which upon click takes to the designated pages.

A sidebar titled category with various categories and upon click shows user products under that categories.



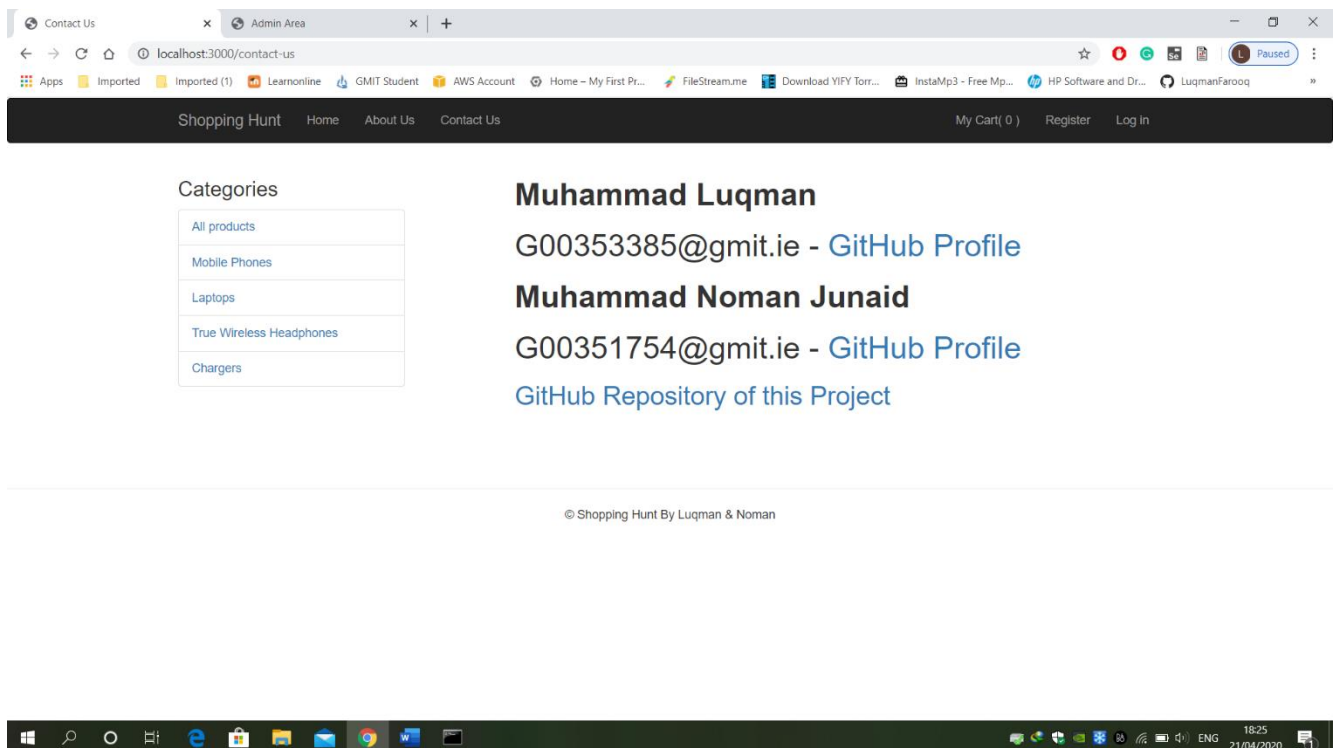
### About Us

Shows all the content described in the homepage just instead of homepage pic it shows about us text.



## Contact Us

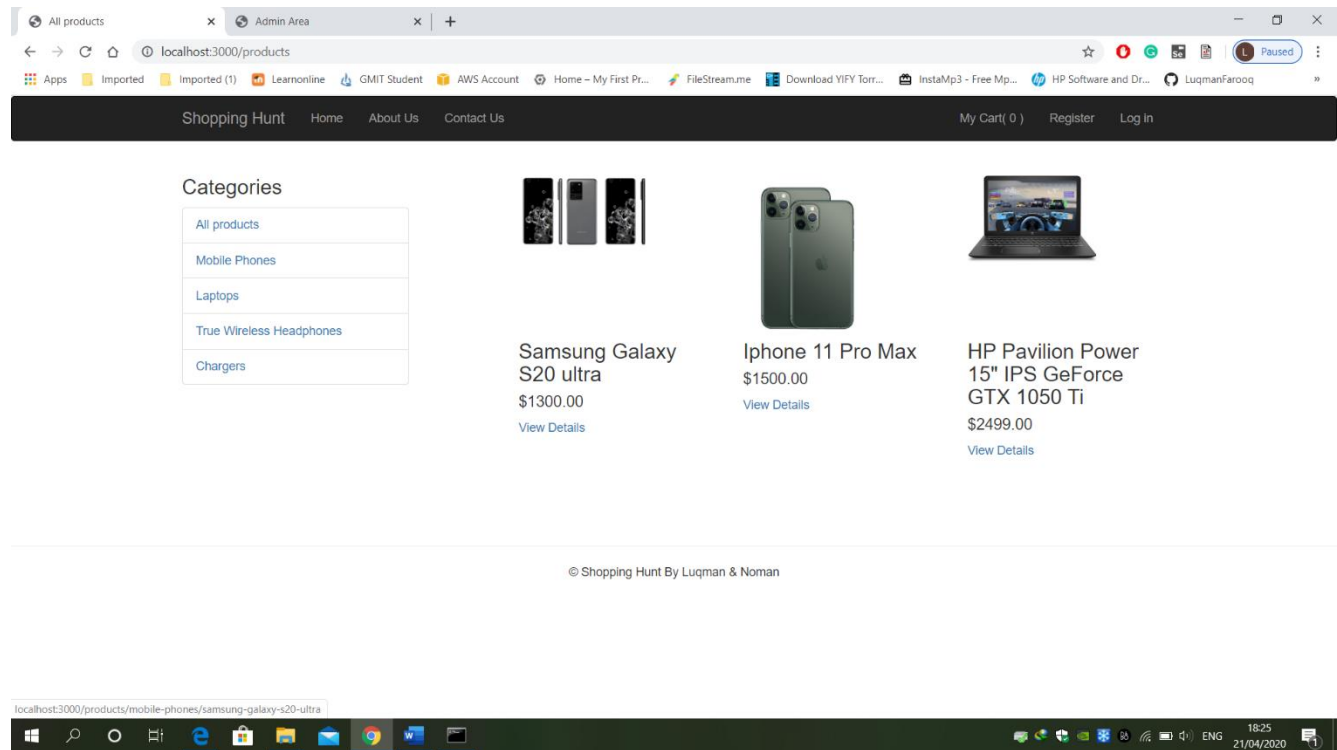
Shows all the content described in the homepage just instead of homepage pic it shows contact us text.





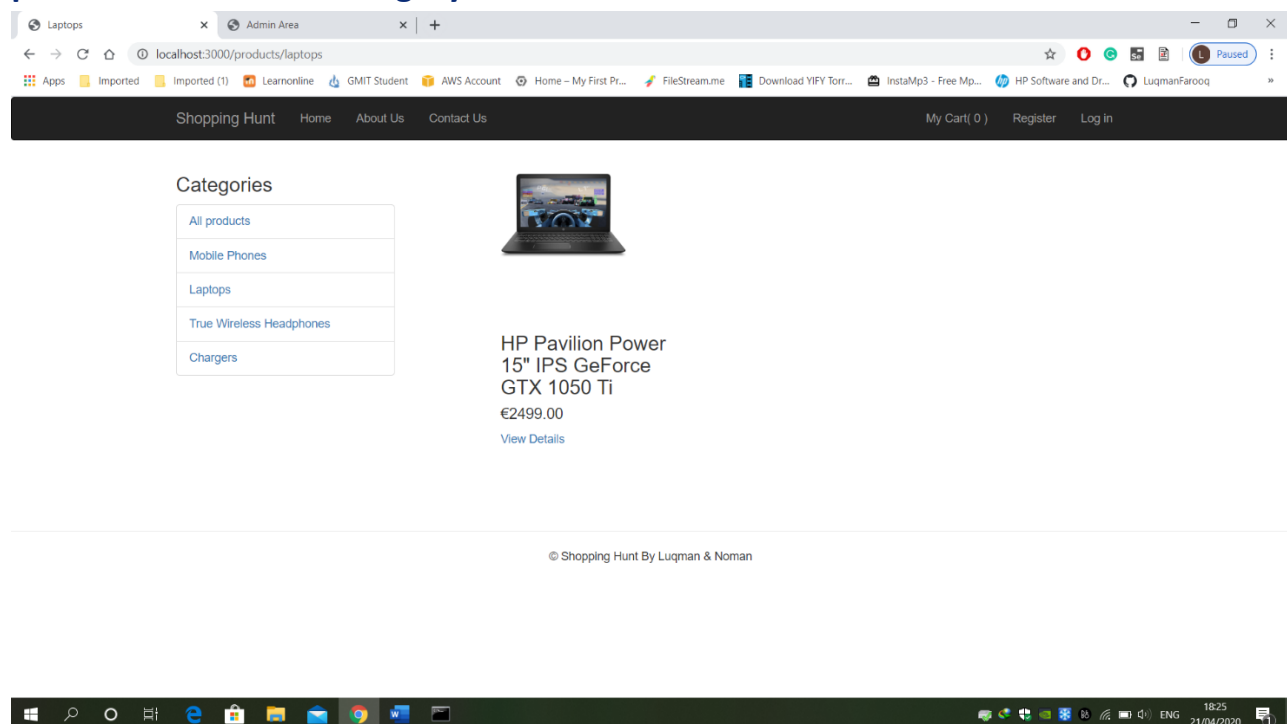
# All Products

It shows all the products and when clicked on any take to the product detail page.



# Category

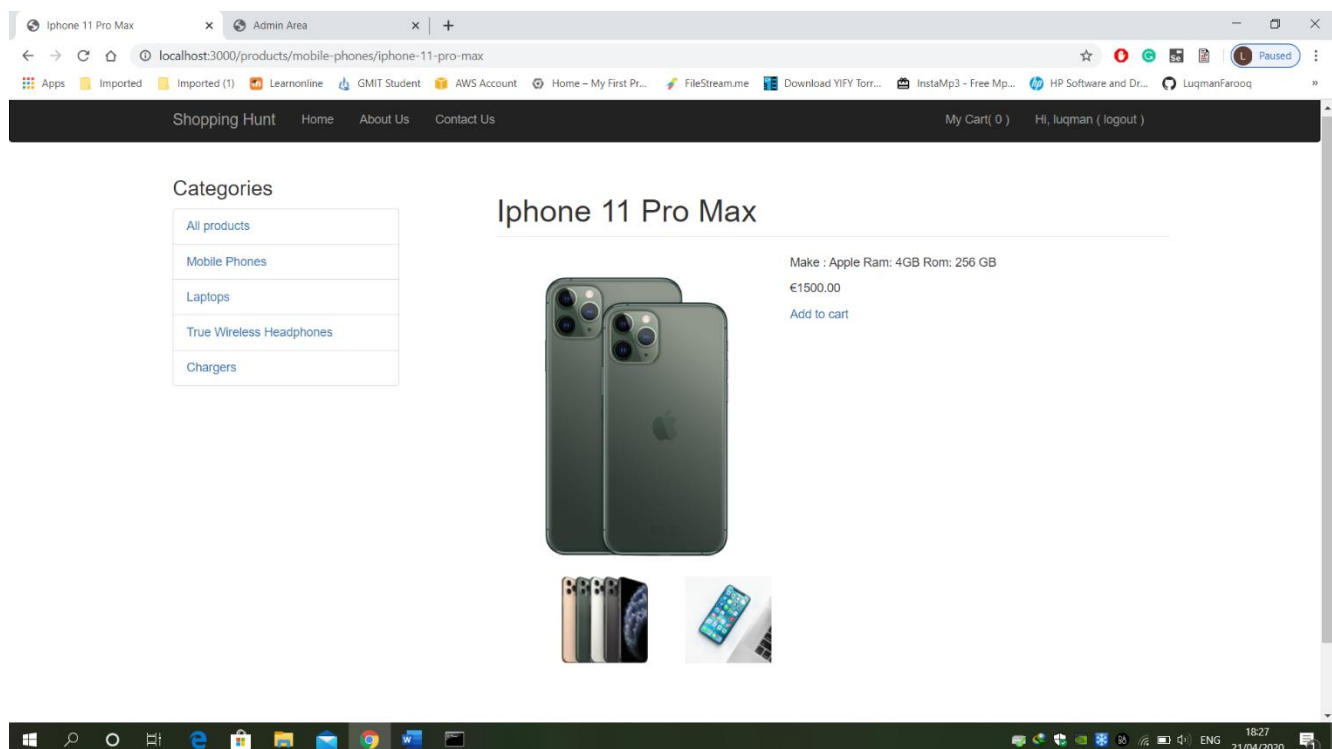
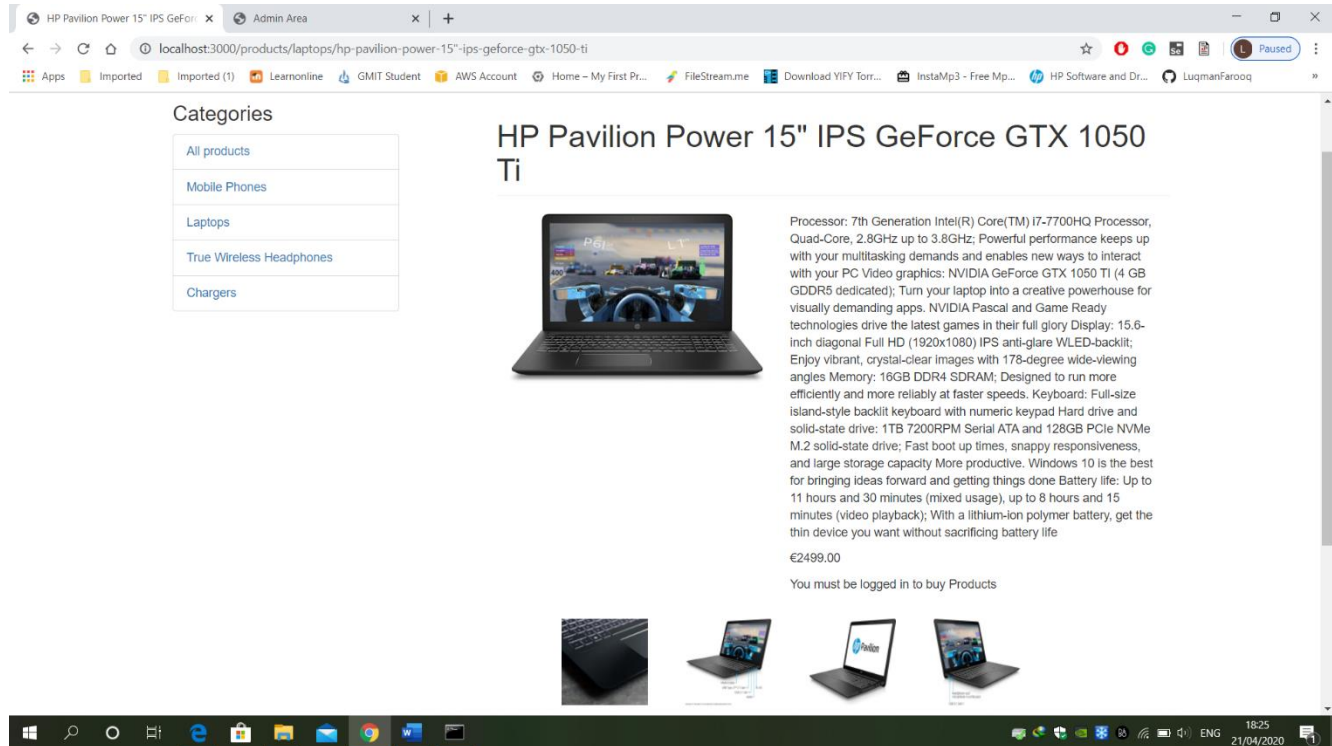
When click on any category it takes the user to that category page which shows products under that category.

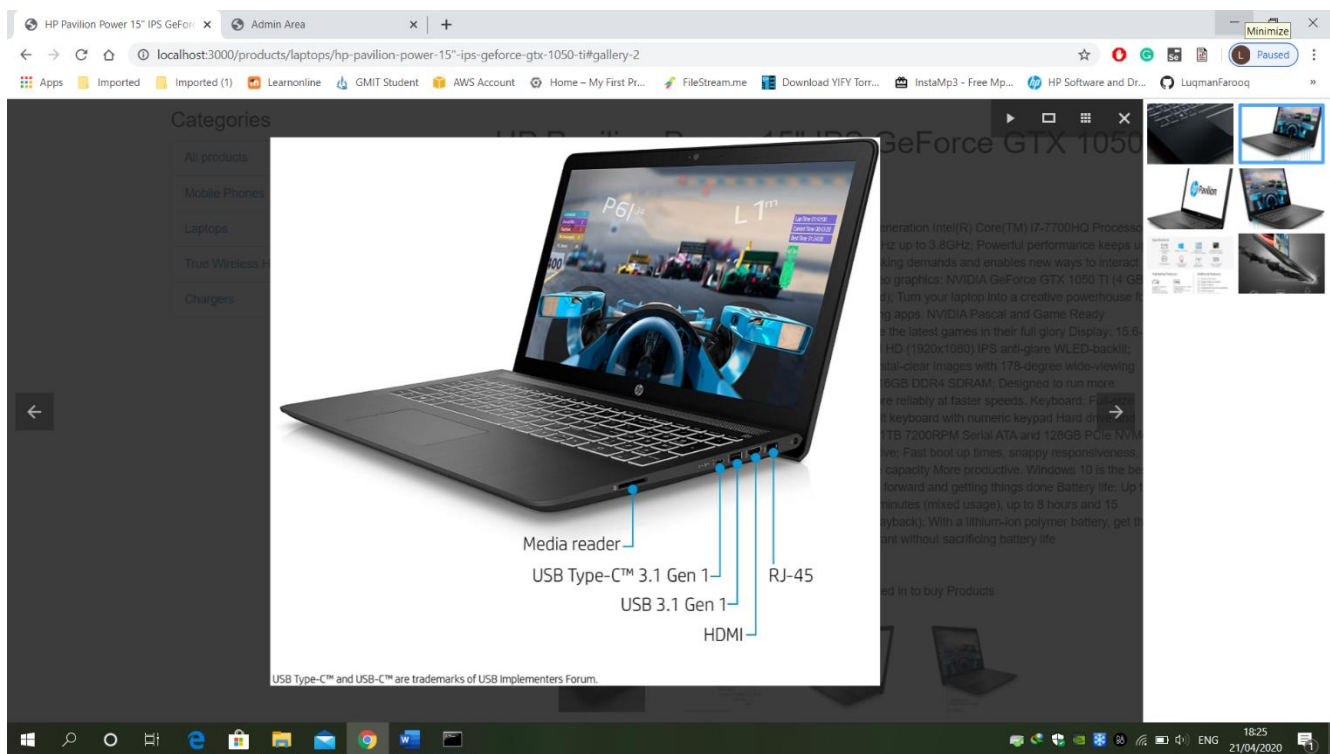


## Product Detail page

The product detail page shows the product details with add to cart button if the user is logged in and “you must be logged in to buy products” text if the user is not logged in.

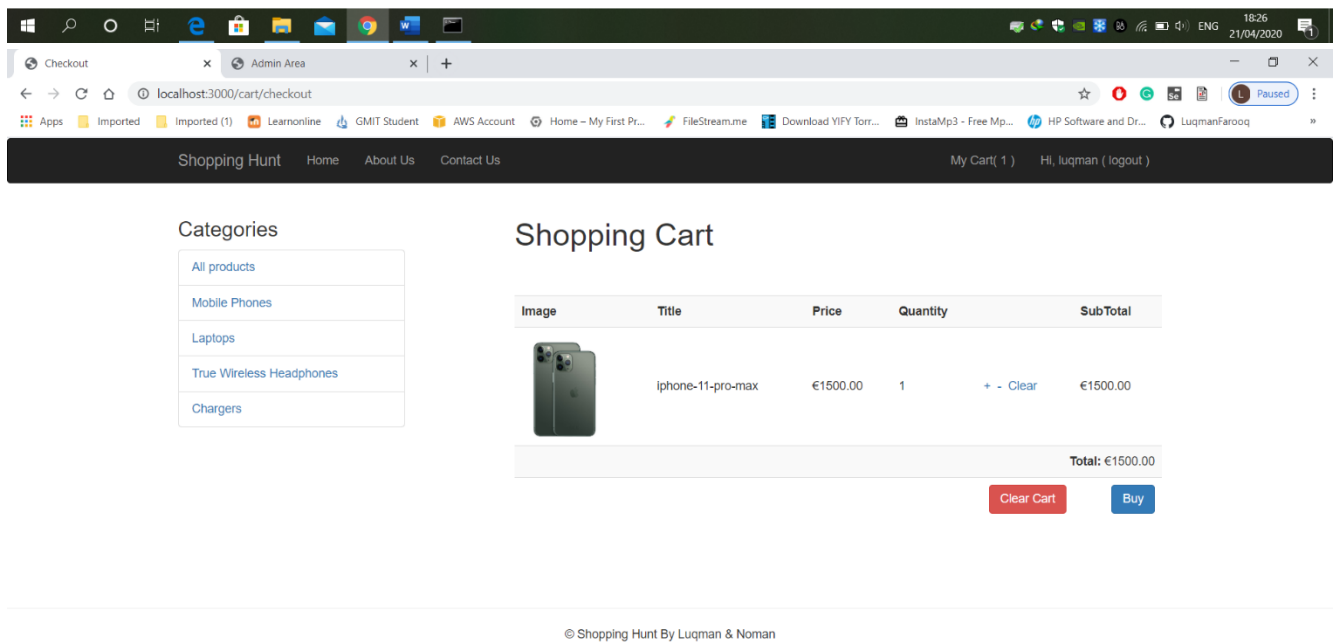
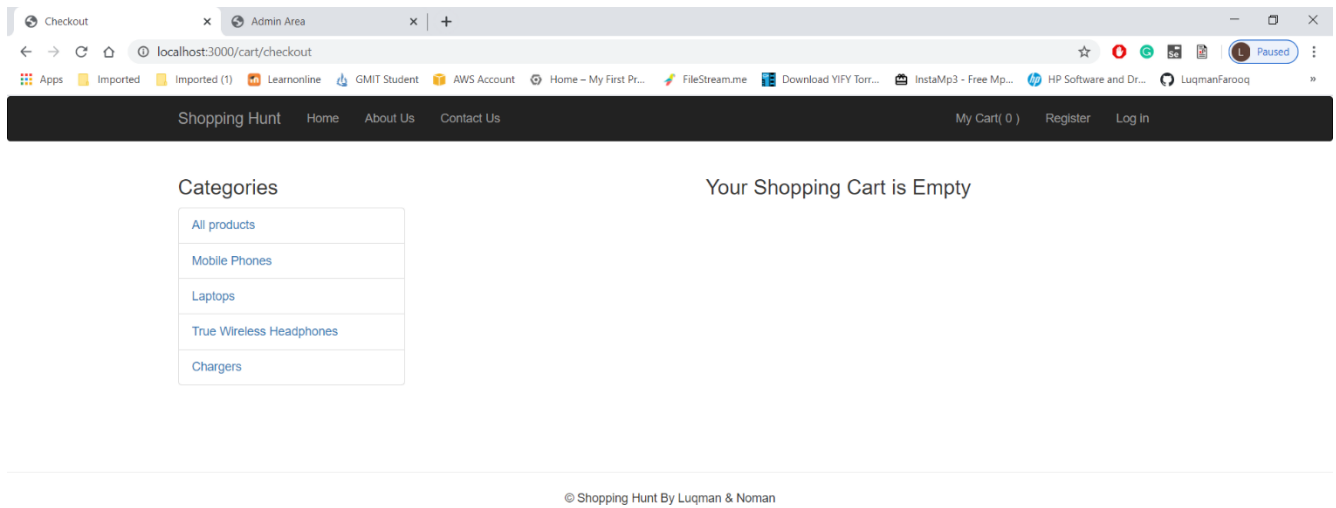
User can click on any product image I opens the pic gallery slider with controls.

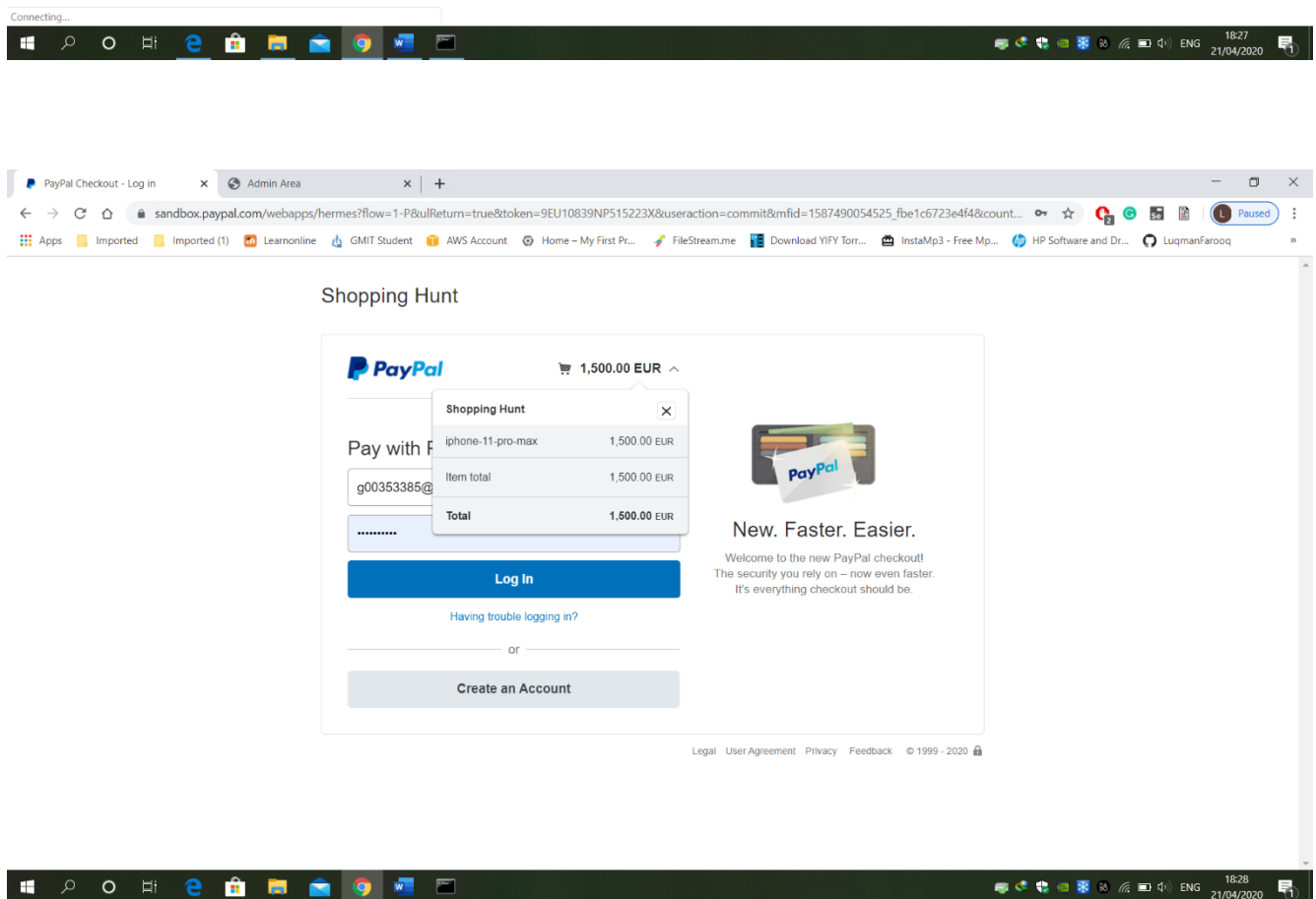
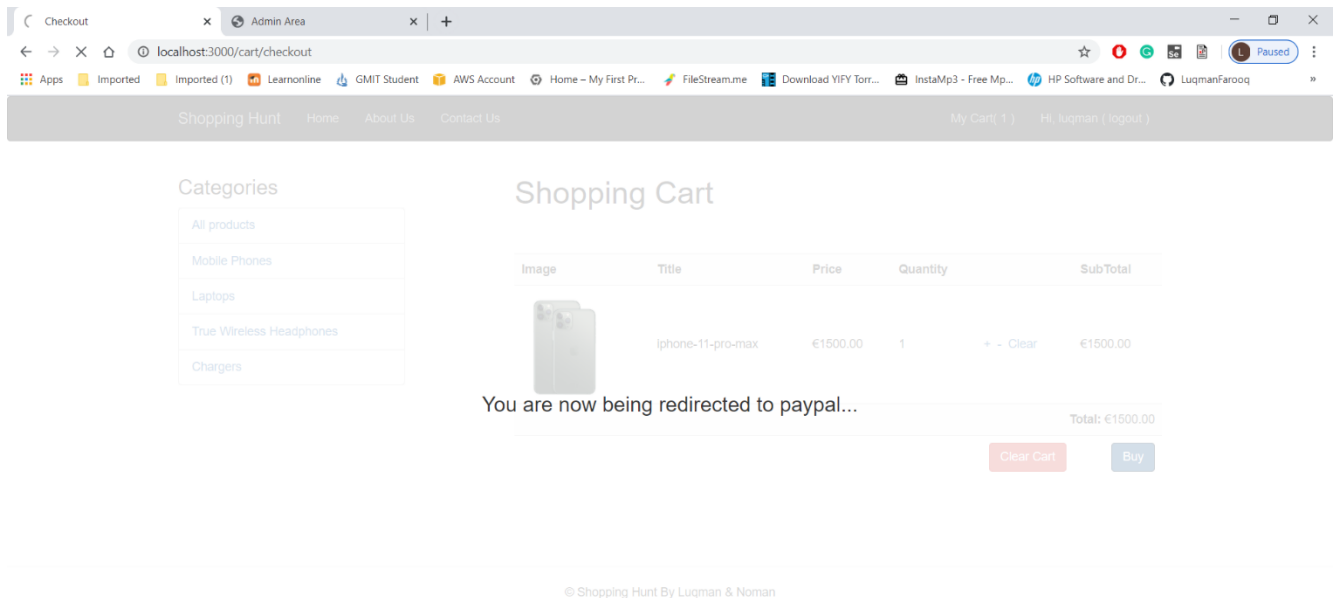




## My Cart

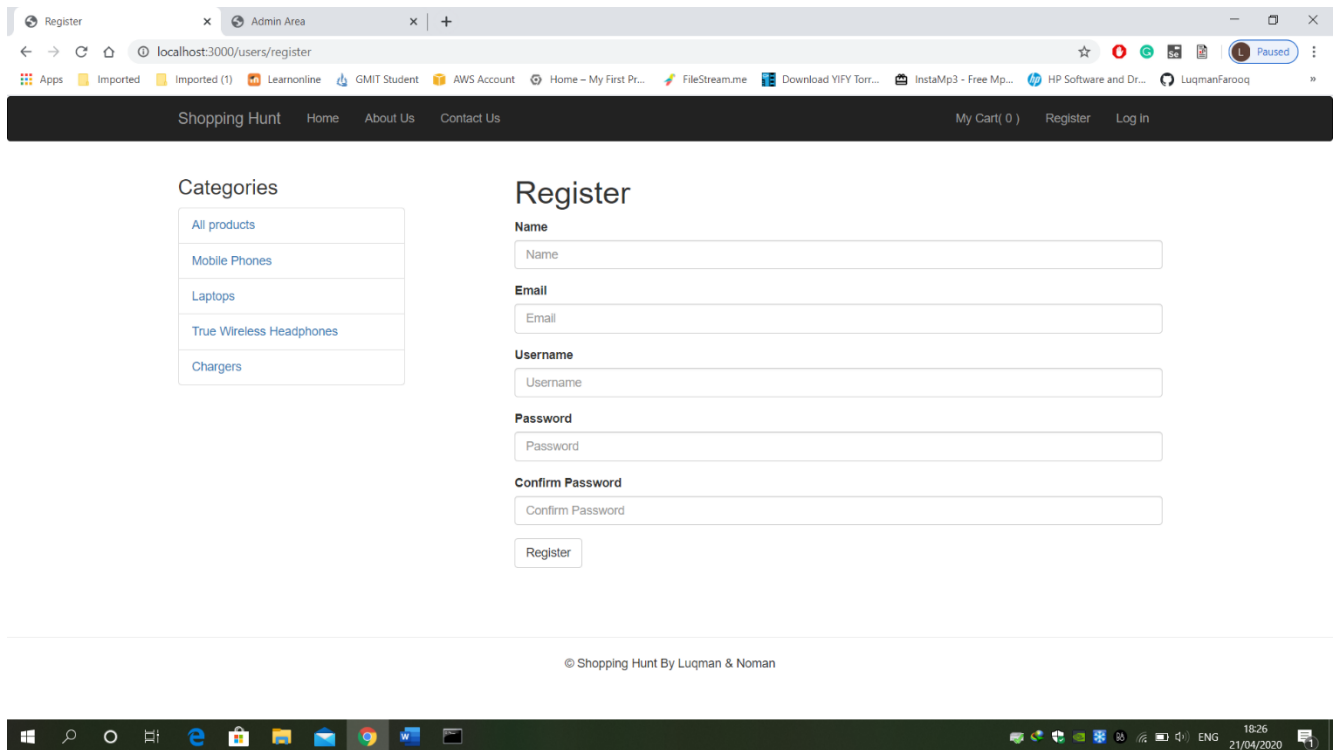
It shows “Your shopping cart is empty” if the user has no item added in the cart. And it shows product pic, price, quantity, clickable buttons such as + to add 1 on each click, - to subtract 1 on each click, clear to clear the cart and buy button which redirects to PayPal to buy product.





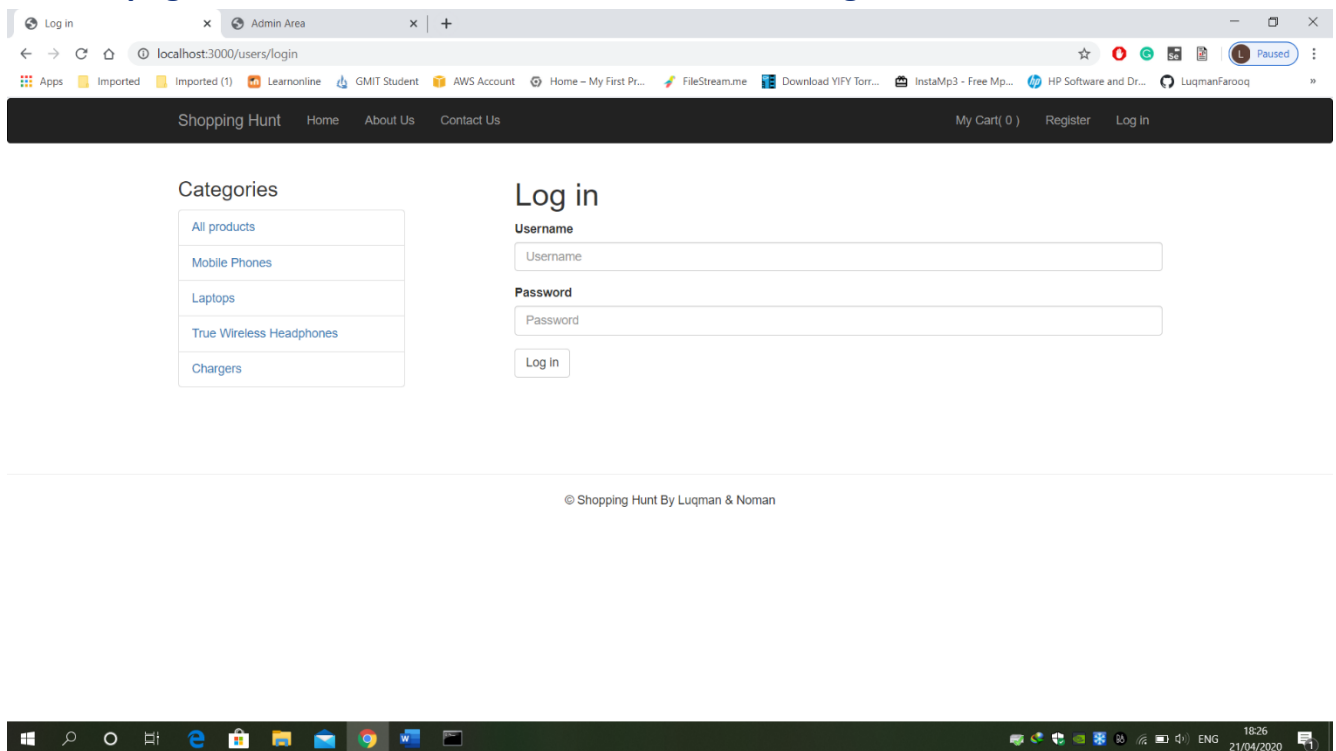
## Register

On this page, the user can enter his details for registration.



## Login

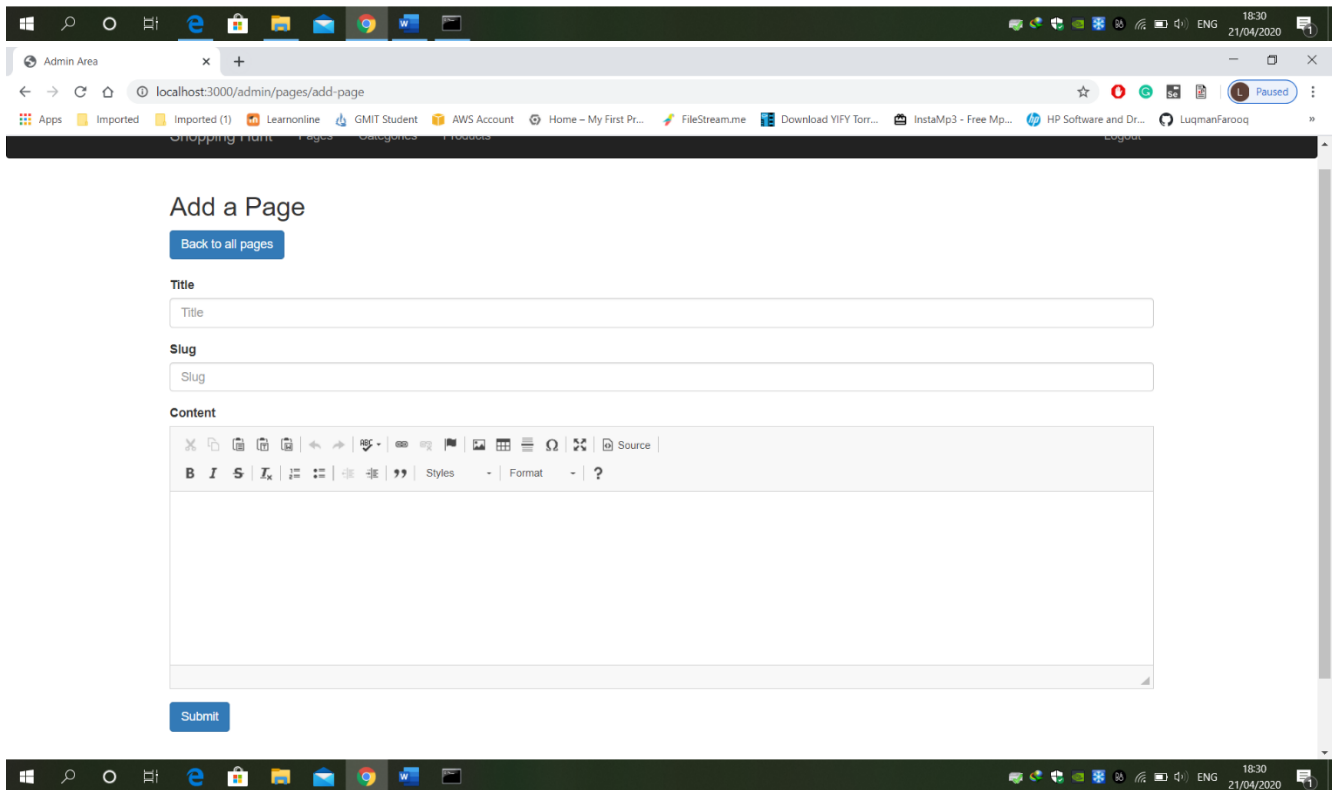
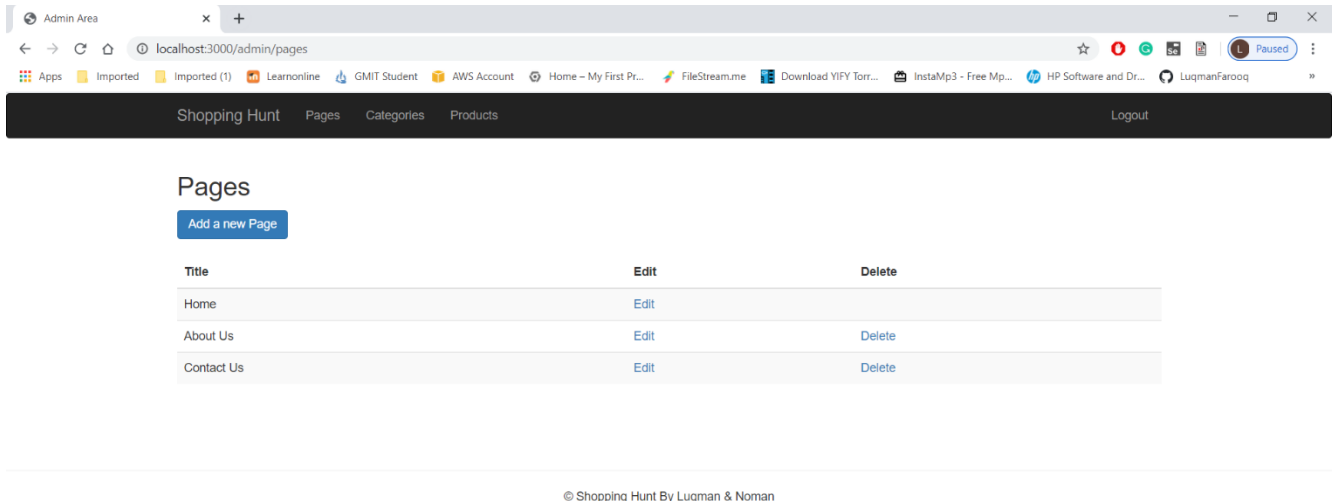
On this page, the user can enter their credentials to log in

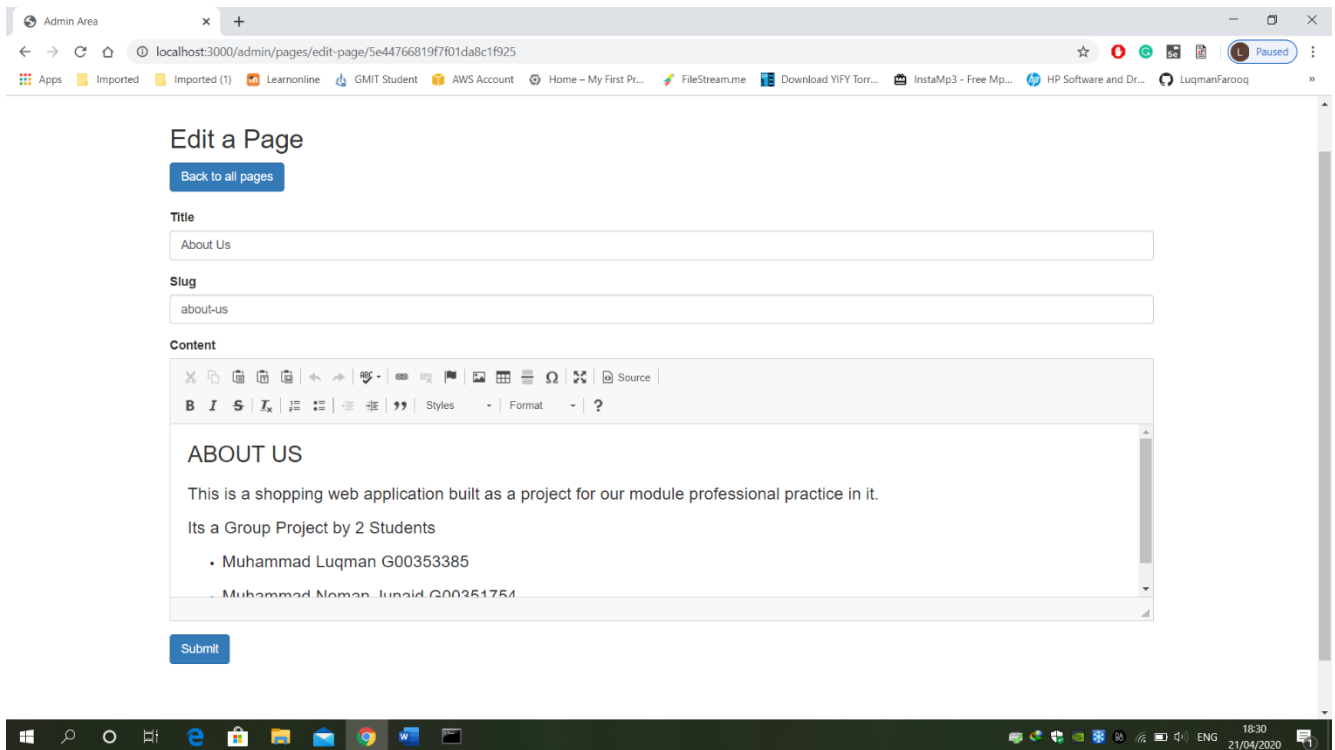


# Admin Area

## Admin Pages

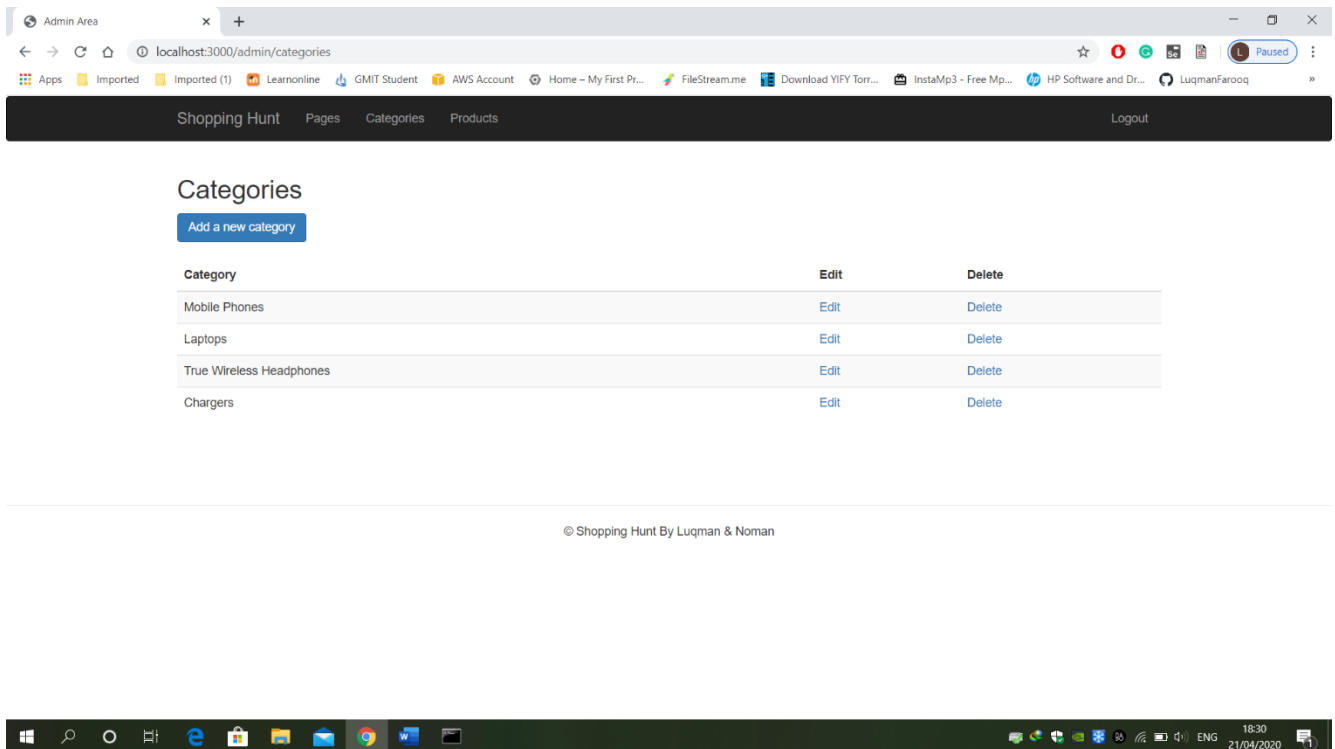
In this page, admin can add pages, edit pages, delete pages, reorder pages the order in which they are shown on navigation bar.



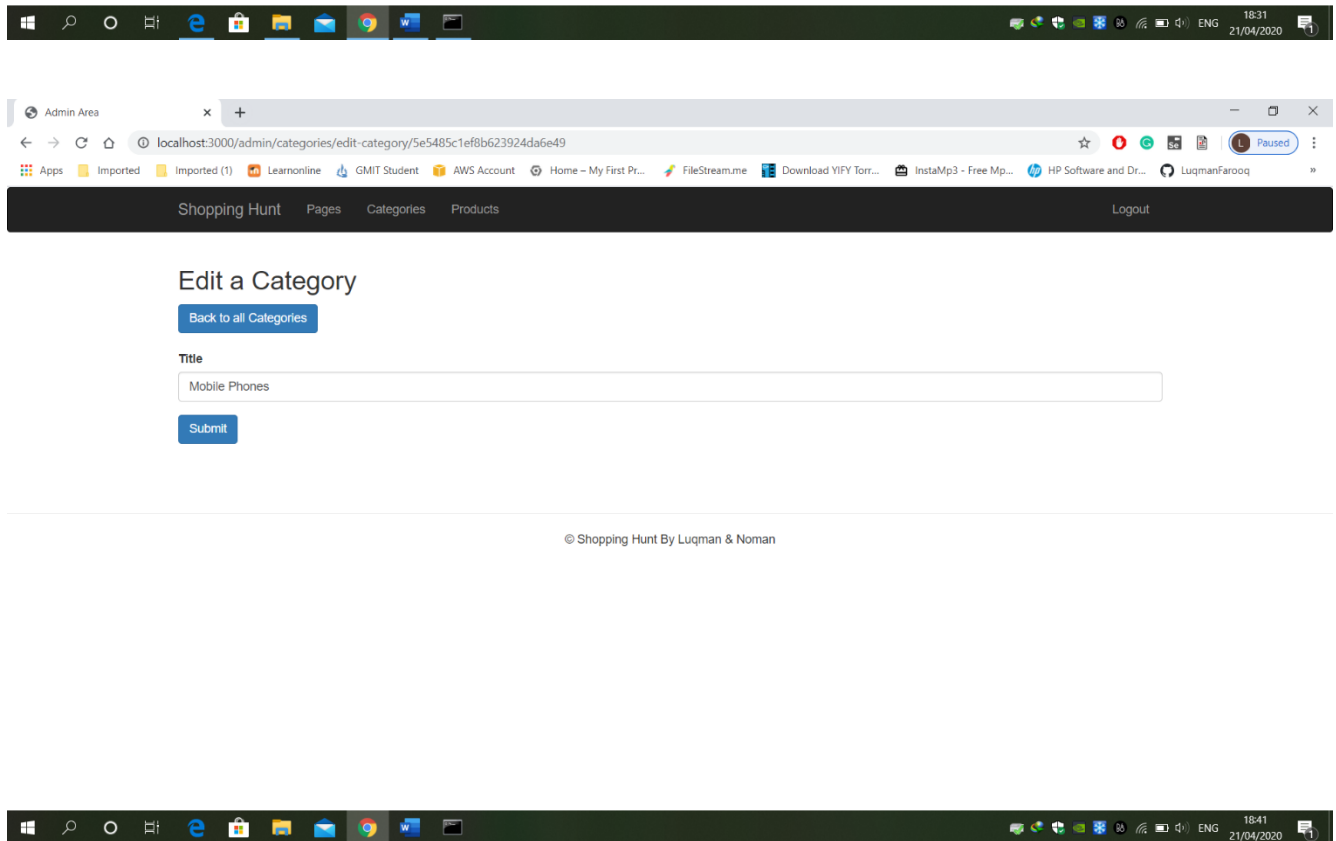
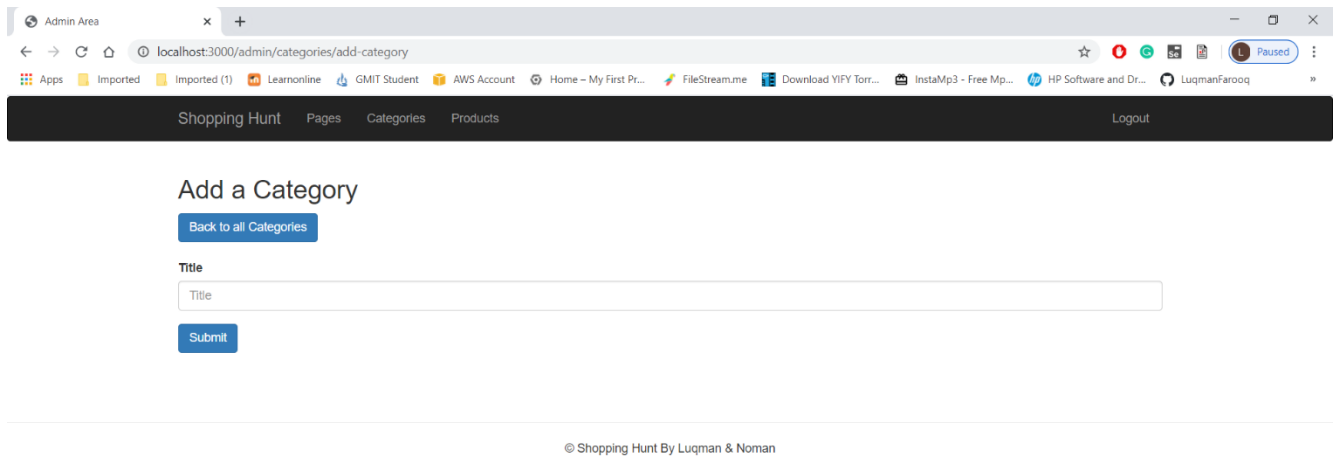


## Admin Categories

on this page admin can add Categories, edit Categories, delete Categories







## Admin Product

On this page, admin can add Product, edit Product, delete Product.

On edit product page admin can add more product images in the product image gallery.




Admin Area

localhost:3000/admin/products

Shopping Hunt Pages Categories Products Logout

## Products

Add a new product

Product	Price	Category	Product Image	Edit	Delete
Samsung Galaxy S20 ultra	€1300.00	mobile-phones		<a href="#">Edit</a>	<a href="#">Delete</a>
Iphone 11 Pro Max	€1500.00	mobile-phones		<a href="#">Edit</a>	<a href="#">Delete</a>
HP Pavilion Power 15" IPS GeForce GTX 1050 Ti	€2499.00	laptops		<a href="#">Edit</a>	<a href="#">Delete</a>

© Shopping Hunt By Luqman & Noman

Admin Area

localhost:3000/admin/products/edit-product/5e58761a248ab42b404db536

Shopping Hunt Pages Categories Products Logout

## Edit product

[Back to all products](#)

**Title**

Samsung Galaxy S20 ultra

**Description**

Ram:8Gb  
Rom:256 Gb


**Category**

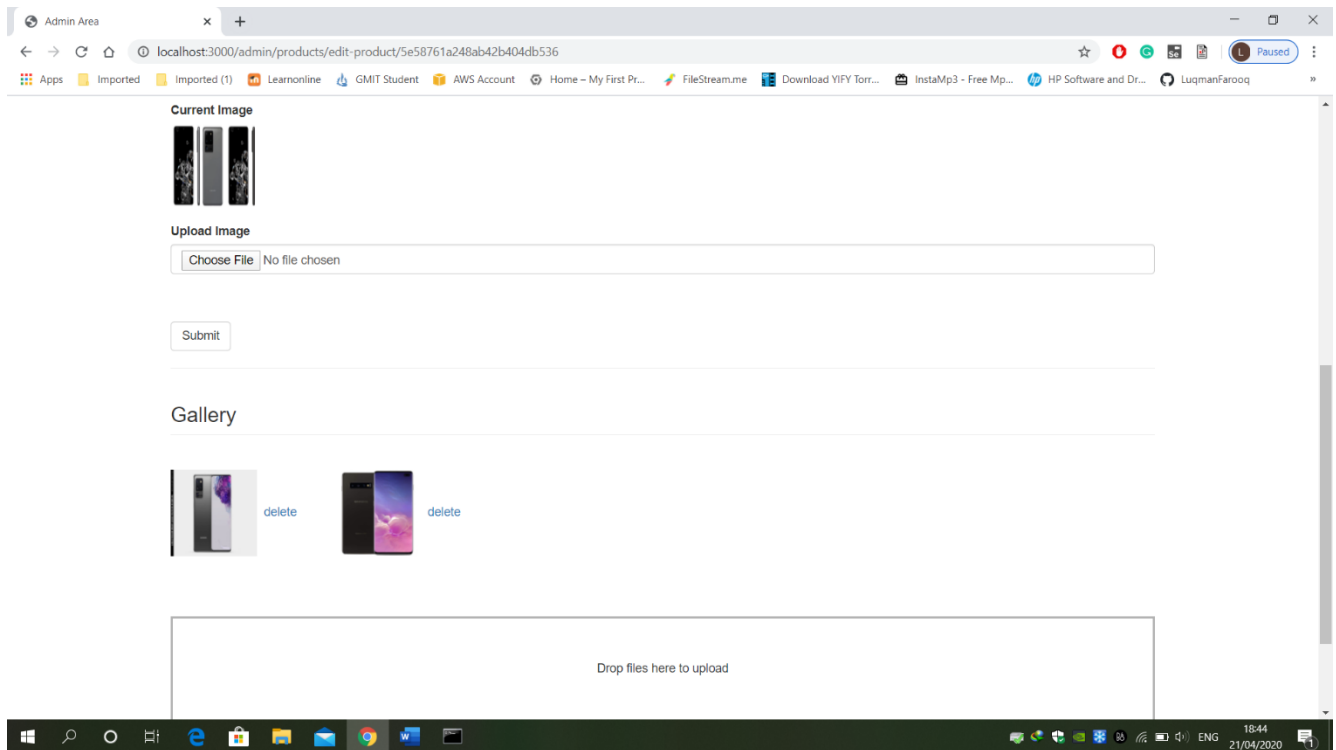
Mobile Phones

**Price**

1300.00

**Current Image**





# System Requirements

## Hardware Requirements

The selection of hardware is very important in the existence and proper working of any software. When selecting hardware, the size and requirements are also important.

**Processor: Intel CORE i5**

**RAM: 4.0 GB**

**Hard Disk Drive: 500 GB**

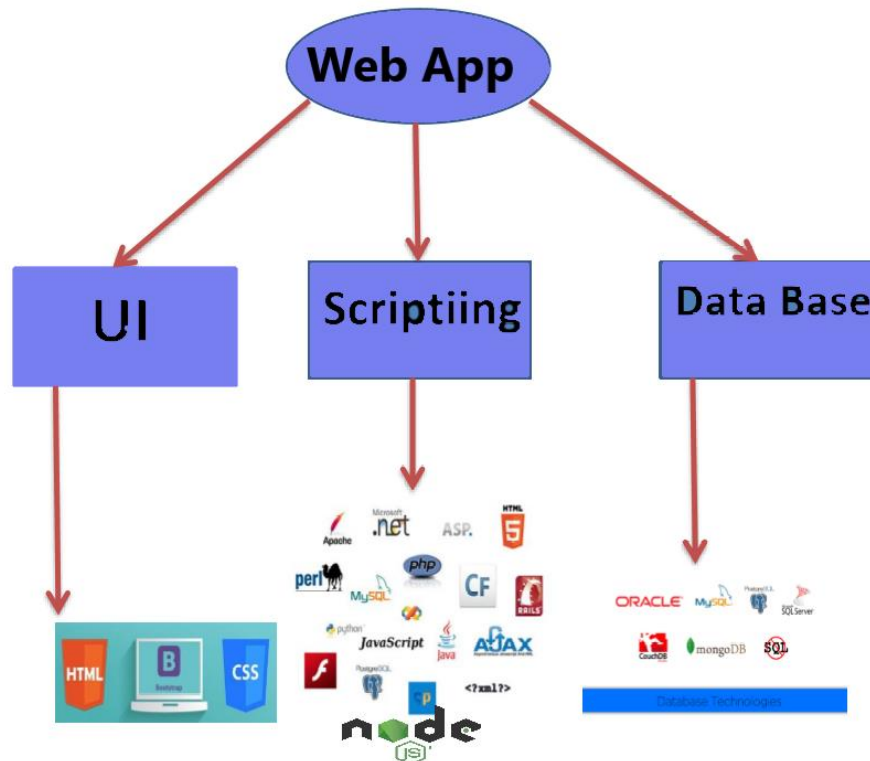
## Software Requirements

1. **Windows 7,8,10**
2. **HTML/CSS/ejs/Ajax/JavaScript/ Bootstrap.**
3. **MongoDB/Mongoose.**
4. **Browser.**

# System Architecture

Creating a web App requires multiple steps which includes the following:

- Creating a UI (User interface)
- Scripting (Both at server end and client end)
- Creating a backend or the database



## UI

UI is built using HTML, CSS, bootstrap, and ejs.

## Scripting

**Server-Side Scripting**

NodeJS is used as server-side scripting.

**Client-Side Scripting**

JavaScript is used for client-side scripting.

## DataBase

Mongo DB is used as a database.

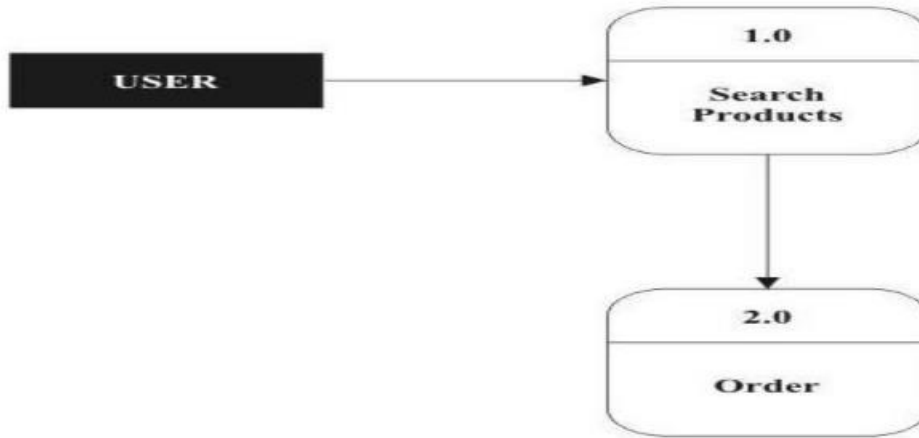
- Category Schema is created for Product categories.

- Page Schema is created for all web app pages.
- Product Schema is created for Products.
- User Schema is created for storing users for registration and login.

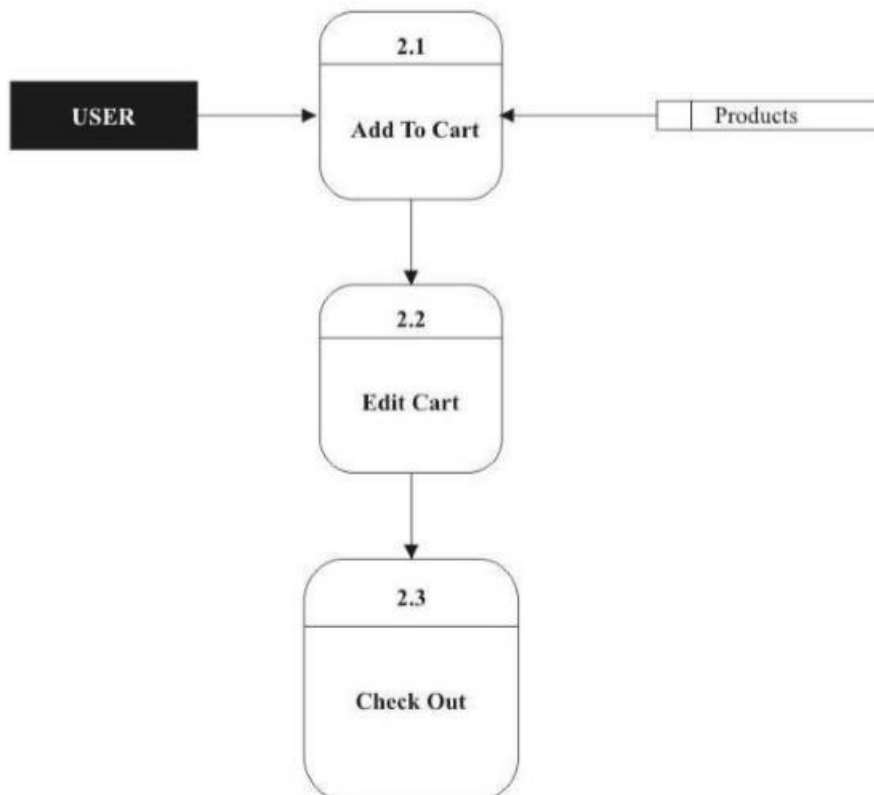
## Data Flow Diagrams



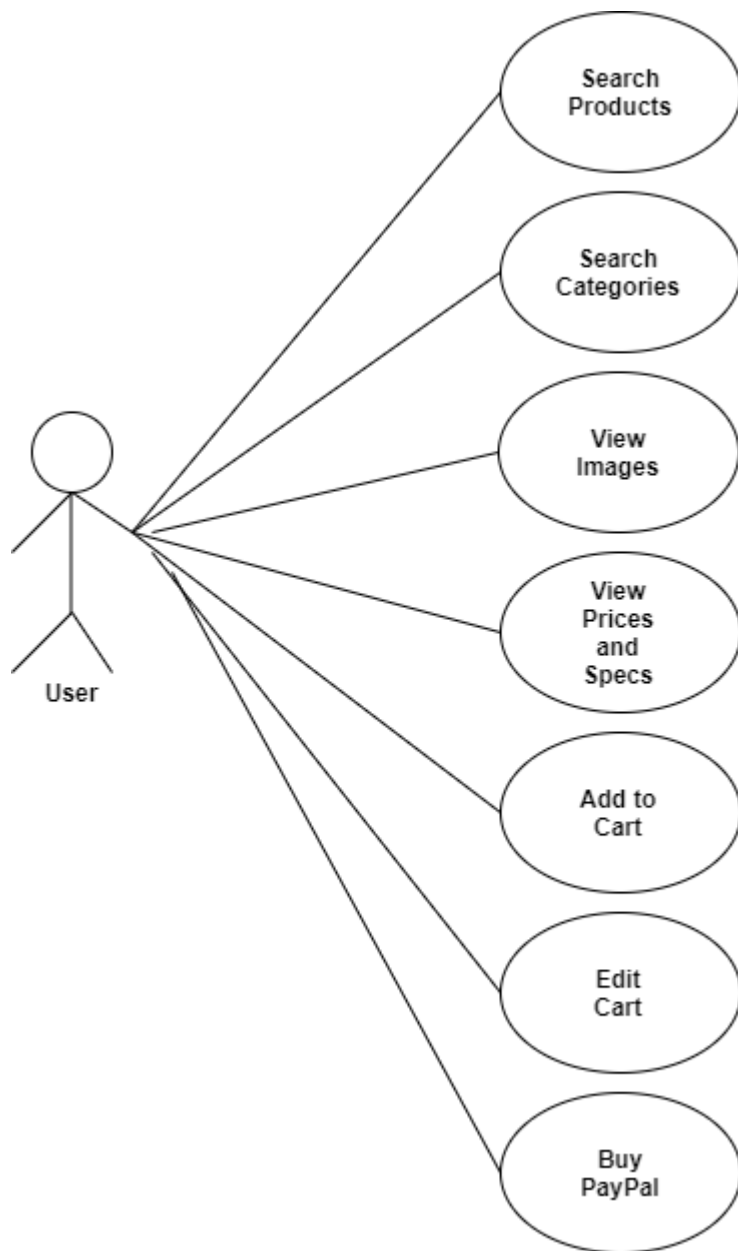
### First Level DFD



### SECOND LEVEL DFD



## Use case Diagram



## Technology Used and Why

- Node
- Express
- EJS templating

- MongoDB
- Mongoose
- Ajax
- Session storage
- Authentication
- Bootstrap
- Html
- CSS

## Node

Node.js is a JavaScript runtime environment built on Chrome's V8 JavaScript engine. It's worth noting that Ryan Dahl, the creator of Node.js, was aiming to create real-time websites with push capability, "inspired by applications like Gmail". In Node.js, he gave developers a tool for working in the non-blocking, event-driven I/O paradigm.

In one sentence: Node.js shines in real-time web applications employing push technology over web sockets.

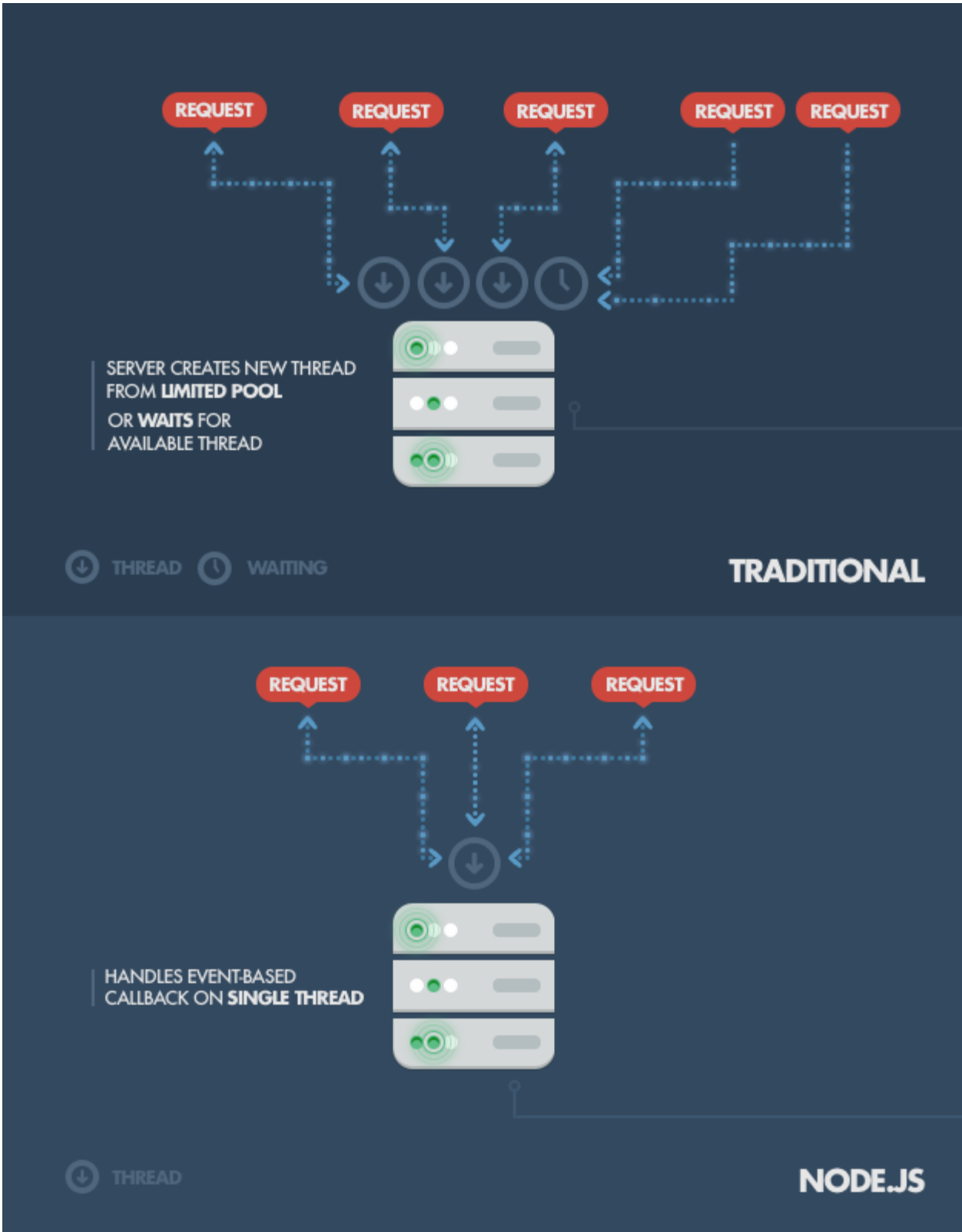
This is in stark contrast to the typical web response paradigm, where the client always initiates communication. Additionally, it's all based on the open web stack (HTML, CSS, and JS) running over the standard port 80.

### How Does It Work?

The main idea of Node.js: use non-blocking, event-driven I/O to remain lightweight and efficient in the face of data-intensive real-time applications that run across distributed devices.

How it works under-the-hood is interesting. Compared to traditional web-serving techniques where each connection (request) spawns a new thread, taking up system RAM and eventually maxing-out at the amount of RAM available, Node.js operates on a single-thread, using non-blocking I/O calls, allowing it to support tens of thousands of concurrent connections (held in the event loop).





## Why Node.js

### SERVER-SIDE WEB APPLICATIONS

Node.js with Express.js can also be used to create classic web applications on the server-side. However, while possible, this request-response paradigm in which Node.js would be carrying around rendered HTML is not the most typical use-case. There are arguments to be made for and against this approach. Here are some facts to consider:

#### Pros:

- If your application doesn't have any CPU intensive computation, you can build it in JavaScript top-to-bottom, even down to the database level if you use JSON storage Object DB like MongoDB. This eases development (including hiring) significantly.
- Crawlers receive a fully-rendered HTML response, which is far more SEO-friendly than, say, a Single Page Application or a web sockets app run on top of Node.js.

#### Cons:

- Any CPU intensive computation will block Node.js responsiveness, so a threaded platform is a better approach. Alternatively, you could try scaling out the computation (\*).
- Using Node.js with a relational database is still quite a pain (see below for more detail). Do yourself a favor and pick up any other environment like Rails, Django, or ASP.Net MVC if you're trying to perform relational operations.

(\*) An alternative to CPU intensive computations is to create a highly scalable MQ-backed environment with back-end processing to keep Node as a front-facing 'clerk' to handle client requests asynchronously.

## Express

ExpressJS is a prebuilt NodeJS framework that can help you in creating server-side web applications faster and smarter. Simplicity, minimalism, flexibility, scalability are some of its characteristics and since it is made in NodeJS itself, it inherited its performance as well.

In short, ExpressJS did for NodeJS what Bootstrap did for HTML/CSS and responsive web design.

It made coding in NodeJS a piece of cake and gave programmers some additional features to extend their server-side coding. ExpressJS is hands down the most famous NodeJS framework- so much so that when most people talk about NodeJS they surely mean NodeJS+ExpressJS.

### Why Express

The first problem new programmers face when developing and launching a web app is a large number of users. If your simple PHP web app, which takes few input parameters and provides small outputs, suddenly is required to take thousands of requests, it would just collapse. NodeJS, on the other hand, will handle it like a boss. Thanks to Google's V8 JavaScript engine. That's what it is powered by.

In addition to lightning-fast JavaScript executions, the real magic behind NodeJS is something called the Event Loop. To scale to large volumes of clients, all I/O intensive operations in NodeJS are performed asynchronously. To avoid inefficiency, NodeJS maintains an Event Loop which manages all asynchronous operations for you. When a NodeJS application needs to perform a blocking operation (I/O operations, heavy computation, etc) it sends an asynchronous task to the Event Loop, along with a callback function, and then continues to execute the rest of its program.

NodeJS is powerful in terms of performance. Now imagine combining this performance with ease of coding. Well, you don't have to imagine it because it's already here in the form of ExpressJS.

Express is preferred because it adds dead-simple routing and support for Connect middleware, allowing many extensions and useful features.

For example,

Want sessions? It's there

Want POST body/query string parsing? It's there

Want easy templating through jade, mustache, ejs, etc? It's there

Want graceful error handling that won't cause the entire server to crash? Yep also built-in

Those are just a few of the features. At its core Express, like the analogous Sinatra for Ruby, adds a powerful and extensible set of features in a cohesive and well-supported whole.

## EJS

EJS simply stands for Embedded JavaScript. It is a simple templating language/engine that lets its user generate HTML with plain JavaScript. It offers an easier way to interpolate (concatenate) strings effectively.

### Why EJS?

Now while there are several templating systems out there with their merits and demerits, I use EJS because of its simplicity in syntax and logic. It's also very easy to set up. It integrates with the Express View Engine.

The default behavior of EJS is that it looks into the 'views' folder for the templates to render.

## MongoDB

Over the years, NoSQL databases such as MongoDB and MySQL have become quite popular as databases for storing data. The ability of these databases to store any type of content and particularly in any type of format is what makes these databases so famous.

Node.js can work with both MySQL and MongoDB as databases. To use either of these databases, you need to download and use the required modules using the Node package manager.

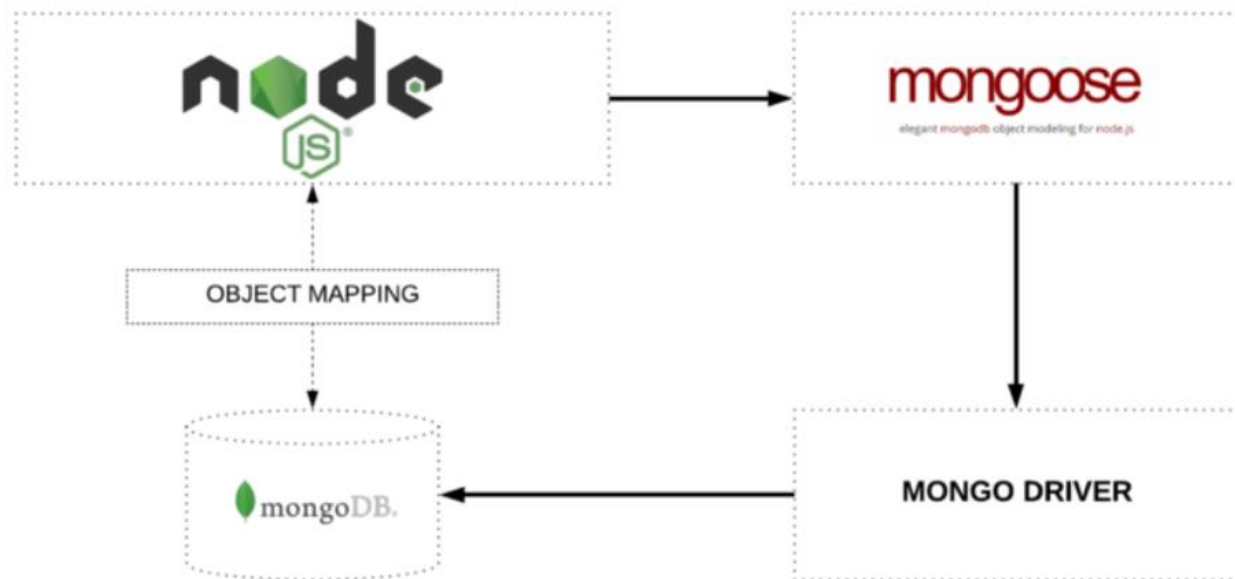
### Why MongoDB

Basically, what made MongoDB so successful is that it was one of the first widely-recognized NoSQL solutions in a time when everybody thought they had "Big Data" and needed a database that was built for scalability first (and integrity last).

Additionally "MEAN" provided a nice buzzword and now the idea that Node.js and MongoDB go together perfectly is stuck in everyone's head and the media (i.e. bloggers and tech websites) perpetuates it because they're too lazy to research to come up with someone more practical.

## Mongoose

Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.



Object Mapping between Node and MongoDB managed via Mongoose

using MongoDB the required module to be installed is "Mongoose."

With these modules, you can perform the following operations in Node.js

- Manage the connection pooling – Here is where you can specify the number of MySQL database connections that should be maintained and saved by Node.js.

- Create and close a connection to a database. In either case, you can provide a callback function that can be called whenever the "create" and "close" connection methods are executed.
- Queries can be executed to get data from respective databases to retrieve data.
- Data manipulation, such as inserting data, deleting, and updating data can also be achieved with these modules.

## Why Mongoose

The main advantage is abstraction over pure mongo.

Many developers who come from SQL database types feel very uncomfortable working with dynamic collections that have no structure defined. So Schemas in the first place helps with that.

Additionally, it implements validation and other neat features to make sure your schema is consistent when inserting/updating/finding documents from collections.

It also creates Model abstraction which makes it easier to work with, so it looks like you are working with just objects rather than pure data.

There are many other goodies like middleware, plugins, population, validation.

Biggest Pro is that it has the data validation built into it (requirements of what data you will allow to be added or to update your database).

It will abstract away most of the mongo DB code from the rest of the application.

## AJAX

AJAX is about updating parts of a web page, without reloading the whole page.

What Is AJAX?

AJAX = Asynchronous JavaScript and XML.

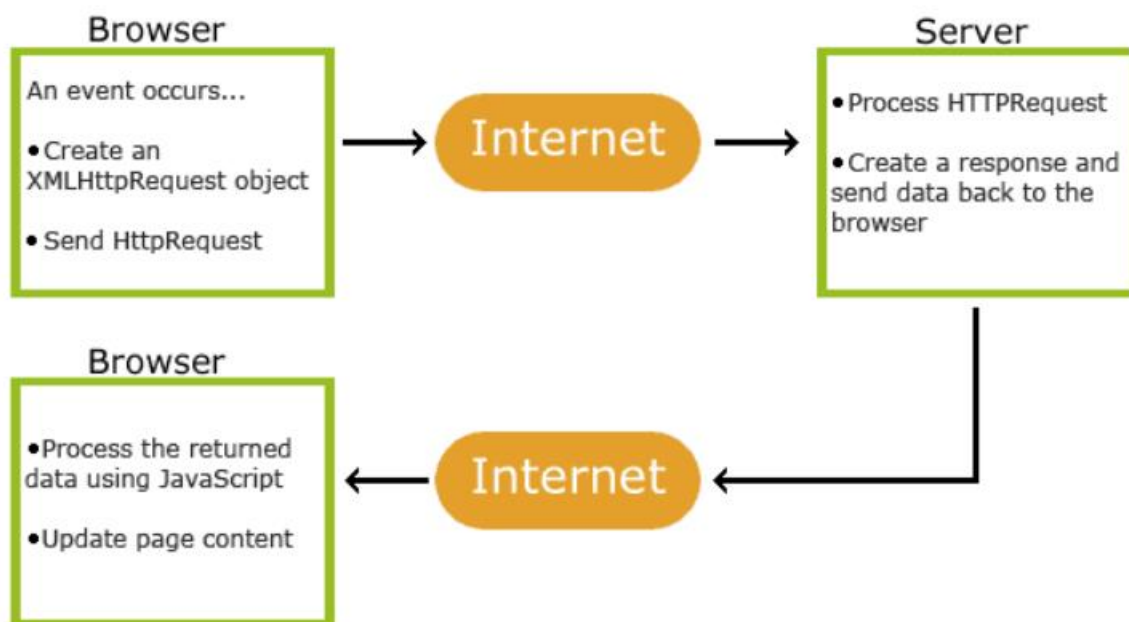
AJAX is a technique for creating fast and dynamic web pages.

AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.

Examples of applications using AJAX: Google Maps, Gmail, YouTube, and Facebook tabs.

### How Ajax Works?



- 1. An event occurs in a web page (the page is loaded; a button is clicked)
- 2. An XMLHttpRequest object is created by JavaScript
- 3. The XMLHttpRequest object sends a request to a web server
- 4. The server processes the request
- 5. The server sends a response back to the web page
- 6. The response is read by JavaScript
- 7. Proper action (like page update) is performed by JavaScript

### Why Ajax

AJAX is based on internet standards and uses a combination of:

- XMLHttpRequest object (to exchange data asynchronously with a server)
- JavaScript/DOM (to display/interact with the information)
- CSS (to style the data)
- XML (often used as the format for transferring data)
- AJAX applications are browser- and platform-independent

AJAX is a developer's dream because you can:

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

## Session Storage

Session Storage is known as the web storage API. Data can be stored on the client-side by using this API.

- Session Storage is used for storing data on the client-side.
- The maximum limit of data saving in Session Storage is about 5 MB.
- Data in the Session Storage exist till the current tab is open if we close the current tab then our data will also erase automatically from the Session Storage.

Why use Session Storage?

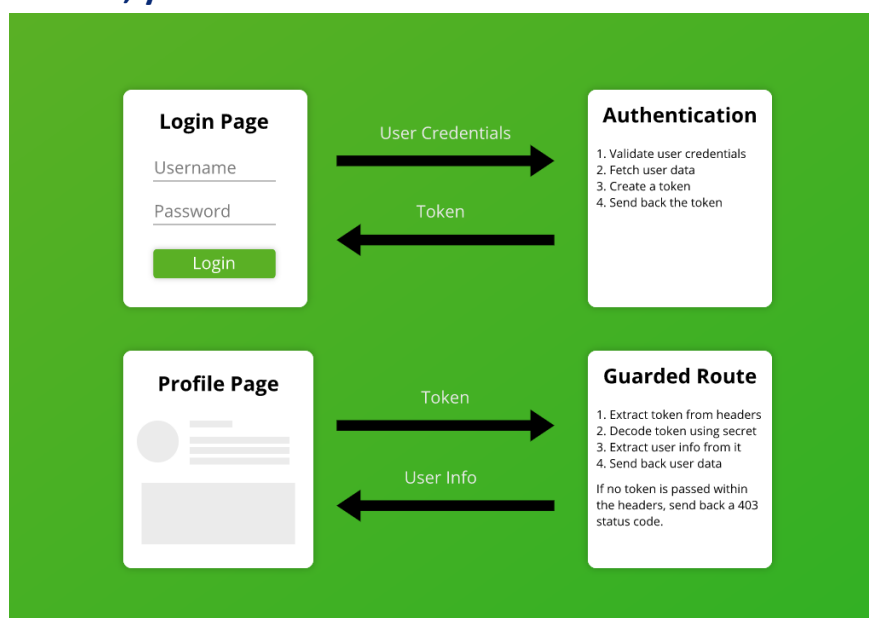
- With ajax-driven dynamic interfaces, a lot of times nothing is storing the current state of how the interface looks (like which tab is selected, for example). session Storage could be used to store the state of the interface, so when coming back to a page, you can restore the screen the way the user was looking at it.
- Another use would be if several pages deep you are working on a single object, you could store the id like a global variable: current Invoice Id.
- User settings that are needed on every page, like a special layout or template, could be loaded once upfront and put into session Storage for easy access.



- Some things you only want the user to see once per login, like a news popup. You could store that they've seen it already in session Storage. This would also work for actions that you only want the user to do once per login.
- It's a good alternative to passing data between pages using view state, hidden `<input>` fields, or URL parameters.
- The main reason to use session Storage is for cases where if your user were to open the same page twice in two different tabs, you'd want separate storage areas for those two tabs. For example, consider a site where you're buying a ticket (and you can only buy one ticket, like an airline ticket flow, as opposed to a case with a shopping cart). If the user tries to buy two tickets in two different tabs, you wouldn't want the two sessions interfering with each other. session Storage lets you track that session across multiple page loads independently.

## Authentication

Passport is the authentication middleware for Node. It is designed to serve a singular purpose which is to authenticate requests. It is not practical to store user password as the original string in the database, but it is a good practice to hash the password and then store them into the database. But with passport-local-mongoose you don't have to hash the password using the crypto module, passport-local-mongoose will do everything for you. If you use passport-local-mongoose this module will auto-generate salt and hash fields in the DB. You will not have a field for the password, instead, you will have salt and hash.



### **Why salting of password is needed**

If the user simply hashes their password and if two users in the database have the same password, then they'll have the same hash. And if anyone of the passwords is hacked then the hacker can access every account using the same password because users with the same password will have the same hash fields.

So before we hash it, we prepend a unique string. Not a secret, just something unique. so, the hash is completely different for every salt.

## **Bootstrap**

Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation, and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

Bootstrap is the second most-starred project on GitHub, with more than 107,000 stars and 48,000 forks.

Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools. Before Bootstrap, various libraries were used for interface development, which led to inconsistencies and a high maintenance burden.

### **Why Bootstrap**

#### **1. Easy to Use**

It is extremely an easy and speedy procedure, to begin with, Bootstrap. Bootstrap is very adaptable too. You can utilize Bootstrap along with CSS, or LESS, or also with Sass [after you download the Sass version].

#### **2. Responsiveness**

Every year mobile devices persist to grow hugely popular, and the requirement to have a responsive website has become compulsory and important too. As the fluid grid layout amends vigorously to the appropriate screen resolution, thus crafting a mobile-ready site is a smooth and easy task along with Bootstrap. With the use of

ready-made classes of Bootstrap, you can recognize the number of spots in the grid system that you would like each column to engage in. Then only you can identify at whichever point you would like your columns to load in horizontal position, instead of vertically to exhibit accurately on mobile appliances.

### 3. The Speed of the Development

One of the main benefits of utilizing Bootstrap happens to be the speed of the development. While driving out a new, fresh website or application swiftly, you should certainly reflect upon utilizing Bootstrap. Instead of coding from scrape, Bootstrap lets you to use ready-made coding blocks to assist you in setting up. You can blend that along with CSS-Less functionality and cross-browser compatibility that can give way to saving of ample hours of coding. You can even buy ready-made Bootstrap themes and alter them to fit your requirements, for gaining the quickest potential route.

### 4. Customizable Bootstrap

The Bootstrap can be customized as per the designs of your project. The web developers can choose to select the aspects which are required which can be simply complete by utilizing Bootstrap customize page. You just have to tick off all the aspects that you do not require, such as- Common CSS: typography, code, grid system, tables, buttons, forms, print media styles; Components: input groups, button groups, pager, labels, nav, navbar, badges, pagination; JavaScript components: dropdowns, popovers, modals, tooltips, carousels; Utilities: Responsive utilities, basic utilities. Thus your custom version of Bootstrap is all set for download process.

### 5. Consistency

Few Twitter employees firstly expanded Bootstrap as a framework for boosting the consistency across interior tools. But later the Co-founder Mark Otto after understanding the actual potential released in August 2011 the first open-source version of Bootstrap. He even portrayed how the Bootstrap was enlarged with the use of one core concept- pairing of designers along with developers. Thus Bootstrap became popular on Twitter.

## 6. Support

As Bootstrap holds a big support community, you can be provided with help whenever there comes any problem. The creators always keep the Bootstrap updated. Presently Bootstrap is hosted, expanded, and preserved on the GitHub along with more than 9,000 commits, as well as more than 500 contributors.

## 7. Packaged JavaScript Components

Bootstrap approaches with a pack of JavaScript components for including the functionality that crafts it in a simple way for operating things, such as tooltips, modal windows, alerts, etc. You can even leave out the writing scripts completely.

## 8. Simple Integration

Bootstrap can be simply integrated along with distinct other platforms and frameworks, on existing sites and new ones too. You can also utilize particular elements of Bootstrap along with your current CSS.

## 9. Grid

Bootstrap can utilize a 12-column responsive grid. It also upholds offset and nested elements. The grid can be maintained in a responsive mode, or you can simply modify it to a secured layout.

## 10. Pre-styled Components

Bootstrap approaches with pre-styled components for alerts, dropdowns, navbars, etc. Hence, being a feature-rich, Bootstrap provides numerous advantages of using it.

## HTML

Hypertext Mark-up Language (HTML) is the standard mark-up language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or local storage and render them into

multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

### Why HTML

- **Web pages development:** HTML is heavily used for creating pages that are displayed on the world wide web. Every page contains a set of HTML tags including hyperlinks which are used for connecting to other pages. Every page that we witness, on the world wide web, is written using a version of HTML code.
- **Web document creation:** Document creation on the internet is dominated by HTML and its basic concept via tag and DOM i.e. document object model. HTML tags are inserted before and afterward or phrases to locate their format and location on the page. A web document consists of three sections: title, head, and body. Head includes the information to identify the document, including title and any other important keyword. A title can be seen on the browser's bar and body section is the main portion of the website visible to the viewer. All three segments are designed and created by the uses of HTML tags. Every section has its own specific set of tags, which are dedicatedly rendered keeping the head, title and body concepts in a loop.
- **Internet navigation:** This is one of the most important uses of HTML which is revolutionary. This navigation is possible by utilizing the concept of Hypertext. It is basically a text which refers to other page and when user click on it, would navigate to referenced text or page. HTML is heavily used to embed the hyperlink within the web pages. A user can easily navigate within the web pages and between websites as well, which are located on different servers.
- **Cutting edge feature:** HTML5 with its set of standards and API is being used to introduce some of the latest trends business. Like polyfill libraries, which are supported by old browsers equally well. Browser like Google Chrome is the perfect choice when it comes to implementing an HTML5 latest set of standards and APIs. There is a javascript library available called Modernizr, which can detect features that let the developer dynamically load polyfill libraries as required.
- **Responsive images on web pages:** At the elementary level in applications of HTML, queries can be set to utilize the images, which are responsive in nature. With the srcset attribute of the image element in HTML, and combining it with

picture element, a developer can fully control how the user will render an image. Now different types of an image with size variation can be loaded by using the image element. Rules can be easily set with the picture element, we can declare image element with default source and then for every case, a source can be provided.

- **Client-side storage:** Earlier, a user could not save the user's browser data that would persist across sessions. To meet this requirement, server-side infrastructure must be built or user's cookies can be used. But with HTML5, client-side storage is feasible using local Storage and Index DB. These two strategies have their own standard and features. local Storage basically gives string-based hash-table storage. Its API is very simple and provides the developer with set Item, get Item, and remove Item methods. Index DB, on the other hand, is a larger and better client-side data store. Index DB database can be expanded with the user's permission.
- **Offline capabilities usage:** Once data can be stored in the browser, the developer can think of a strategy to make application work, when a user is disconnected. HTML5 has its application cache mechanism which would define how the browser manages the offline situation. Application cache, responsible for offline ability actually comprises of different components, which includes API methods that create an update, read manifest file and events. By using certain properties in HTML5, a developer can check if the application is online or not. A developer can also specify in the website's application cache manifest file the information like, what browser manages resources for offline use. In the manifest file, resources which are available offline can also be specified.
- **Data Entry support with HTML:** HTML5 standard and set of APIs can be used to support data entry level of work. As browsers implement new HTML5 standards, developers can simply add the attributes to the tag which indicate required fields, text, data format, etc. HTML5 has come up with several new attributes to drive on-screen keyboards, validation, and other data-entry experiences so that end-user can have a better data-entry.
- **Game development usage:** Before the advent of HTML5, game development was an exclusive domain of Flash and Silverlight. Since browsers support new specifications for HTML5 including CSS3 and light-fast JavaScript engine to drive a new rich experience, HTML5 can bring the reality of game development possible, which was earlier the forte of Flash and Silverlight. Every single

feature of APIs need not be implemented, but most appropriate ones can be utilized while eliminating the rest of the features.

- Native APIs usage to enrich the website: HTML5 adds so many new abilities and tools, which was just an imagination in the past. A large set of new APIs regarding file system, Geolocation, drag, and drop, event handling, client-storage, etc. are the capabilities which make usage of HTML5 easier than ever before. Application experience can be enhanced with other APIs like Fullscreen, Visibility and Media Capture. Source: Educa. Com

## CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a mark-up language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG, and XUL, and applies to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate CSS file and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same mark-up page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display

14

the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

## Why CSS

Because we want to separate the content (HTML) from its presentation (CSS).

If you want to visualize the purpose of this distinction, head towards the wonderful [CSS Zen Garden](#): each design uses the *exact* same HTML but a *different* CSS each time.

It makes maintenance easier as well: the same CSS file can be used for a whole website. It provides flexibility: focus on the content on one side, the styling on the other. SEO purposes, different concerns.

## Limitations and Known Bugs

Admin products page sometimes needs a couple of refreshes before it shows products on that page but on the client page, it is working fluently.

## Testing Plans

- We did unit testing and tested the web app bit by bit as we added pages.
- We did integration testing as we integrated components.
- We did validation testing to validated inputs and validate users' credentials.
- We did complete system testing after completion of the web app.

## Recommendations for Future Development

- To be able to add a video demo of the products. So, the clients would have a more clear perspective of the product.
- To add a direct secure card payment method.
- To save customer details such as delivery address.
- To keep track of sales and products sold more common.
- To generate suggestions based on the customer products view history.

## Conclusions

We have successfully implemented the Web App 'Shopping Hunt'. With the help of various links and tools.

- we have been able to provide a Web App which will be live soon and running on the web.



- We have been successful in our attempt to take care of the needs of both the user as well as the administrator.
- We have learned a new language embedded JavaScript.
- We learned how to implement register and login in node.js.
- we learned how to link PayPal for secure checkout.

Finally, we hope to add more features as described in the future recommendations above and that this will go a long way in popularizing.

## References

<https://scotch.io/tutorials/use-ejs-to-template-your-node-application>

<http://www.bryanmierke.com/2018-02-09-url-slugs-with-nodejs-mongodb>

<https://code.jquery.com/ui/>

[https://medium.com/@Linda\\_Ikechukwu/](https://medium.com/@Linda_Ikechukwu/)

<https-medium-com-linda-ikechukwu-using-ejs-as-a-template-engine-in-your-express-app-cb3d82c15e17>

<https://www.geeksforgeeks.org/use-ejs-as-template-engine-in-node-js/>

<https://www.quora.com/Why-should-I-use-Express-when-developing-a-web-app-with-Node-js>

<https://www.algoworks.com/blog/why-use-expressjs-over-nodejs-for-server-side-coding/>

[https://www.reddit.com/r/node/comments/2vn542/why\\_everybody\\_uses\\_nodejs\\_with\\_mongodb/](https://www.reddit.com/r/node/comments/2vn542/why_everybody_uses_nodejs_with_mongodb/)

<https://www.freecodecamp.org/news/introduction-to-mongoose-for-mongodb-d2a7aa593c57/>

<https://www.guru99.com/node-js-mongodb.html>

<https://stackoverflow.com/questions/18531696/why-do-we-need-what-advantages-to-use-mongoose>

<https://medium.com/the-node-js-collection/why-the-hell-would-you-use-node-js-4b053b94ab8e>

<https://www.geeksforgeeks.org/localstorage-and-sessionstorage-web-storage-apis/>

[https://www.w3schools.com/php/php\\_ajax\\_intro.asp](https://www.w3schools.com/php/php_ajax_intro.asp)

<https://stackoverflow.com/questions/8498357/when-should-i-use-html5-sessionstorage>

<https://www.geeksforgeeks.org/nodejs-authentication-using-passportjs-and-passport-local-mongoose/>

<https://www.quora.com/What-are-the-uses-of-HTML>

<https://marksheet.io/why-css-exists.html>

<https://www.quora.com/What-is-the-use-of-Bootstrap>