ASSIGNMENT 2 FRONT SHEET

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number and title	Unit 9: Cloud Computing		
Submission date		**Date Received 1st submission**	
Re-submission Date		**Date Received 2nd submission**	
Student Name	Nguyen Quoc Anh	**Student ID**	GCH18888
Class	GCH0718	**Assessor name**	
Student declaration I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.			
		Student's signature	

Grading grid

P5	P6	P7	P8	M3	M4	D2	D3

❒ Summative Feedback:	❒ Resubmission Feedback:	
Grade:	Assessor Signature:	Date:
Internal Verifier's Comments:		
Signature & Date:		

Contents

A. INTRODUCTION

In the last report, we have talk about some fundemental concept of cloud computing. Today, we will implement a web application in cloud depend on the solution which i provided in the last report. There is a web application with Nodejs and using mongodb as database. These project will be deploy in internet with Heroku.

B) DESIGN

I. Overview Function

1. Solution

We have chosen cloud computing, after clearly analyzing the needs of businesses as well as the benefits that cloud computing brings. Implementation will be detailed in this report.

2. Use-case diagram

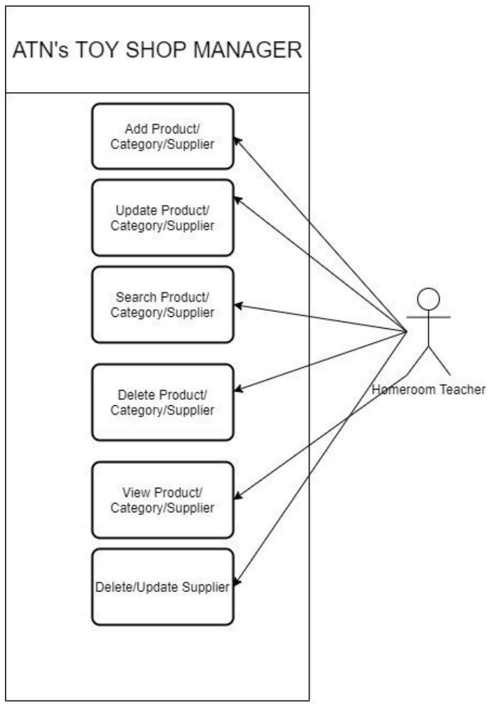

ATN's TOY SHOP MANAGER
Add Product/ Category/Supplier
Update Product/ Category/Supplier
Search Product/ Category/Supplier
Delete Product/ Category/Supplier
View Product/ Category/Supplier
Delete/Update Supplier
Homeroom Teacher

Figure 1: Use case diagram

Web Application has the several function that are:

- Add/Update/Delete/View/Search Product
- Add/Update/Delete/View/Search Category
- Add/Update/Delete/View/Search Supplier

3. Website Screen Shots

3.1 Index

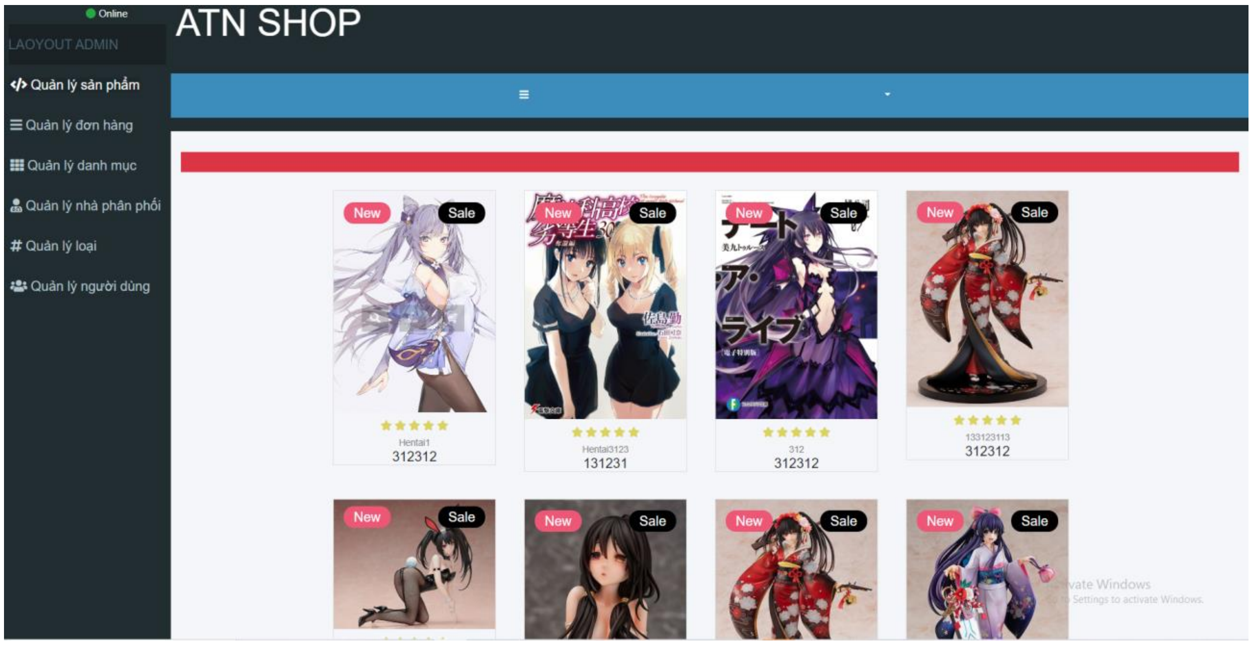
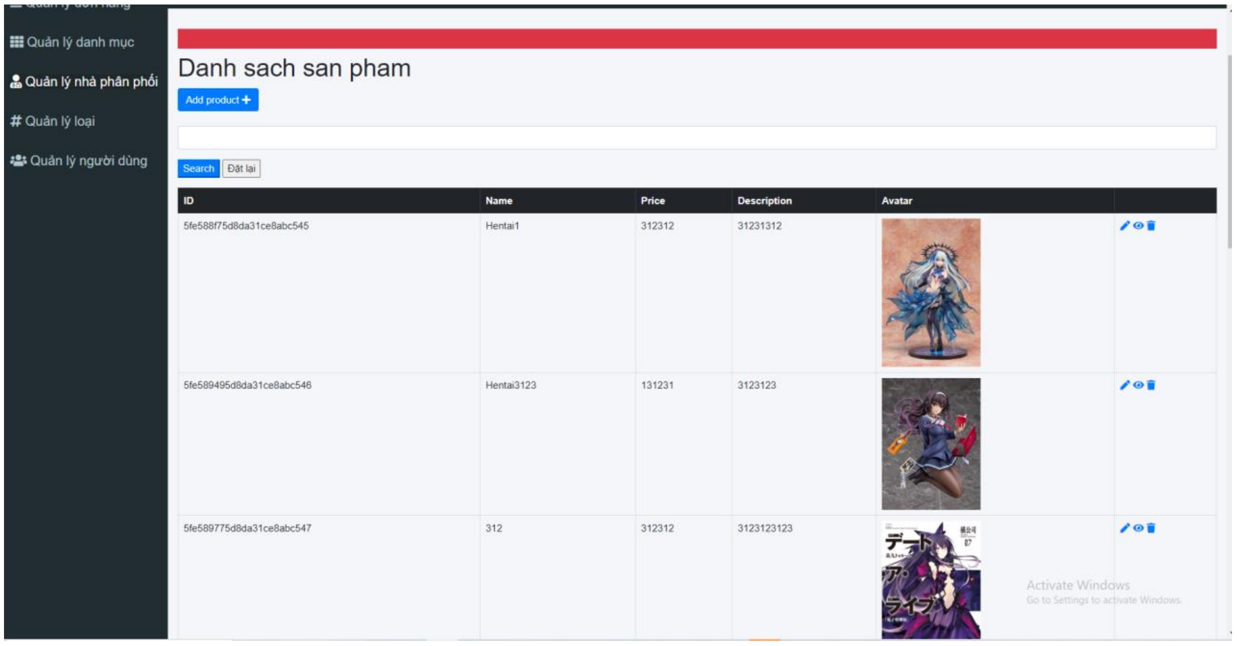


Figure 2: Homepage

This is the first interface when using the website. It will show product and a navbar for menu list of function.

3.2 List Product/Category/Supplier



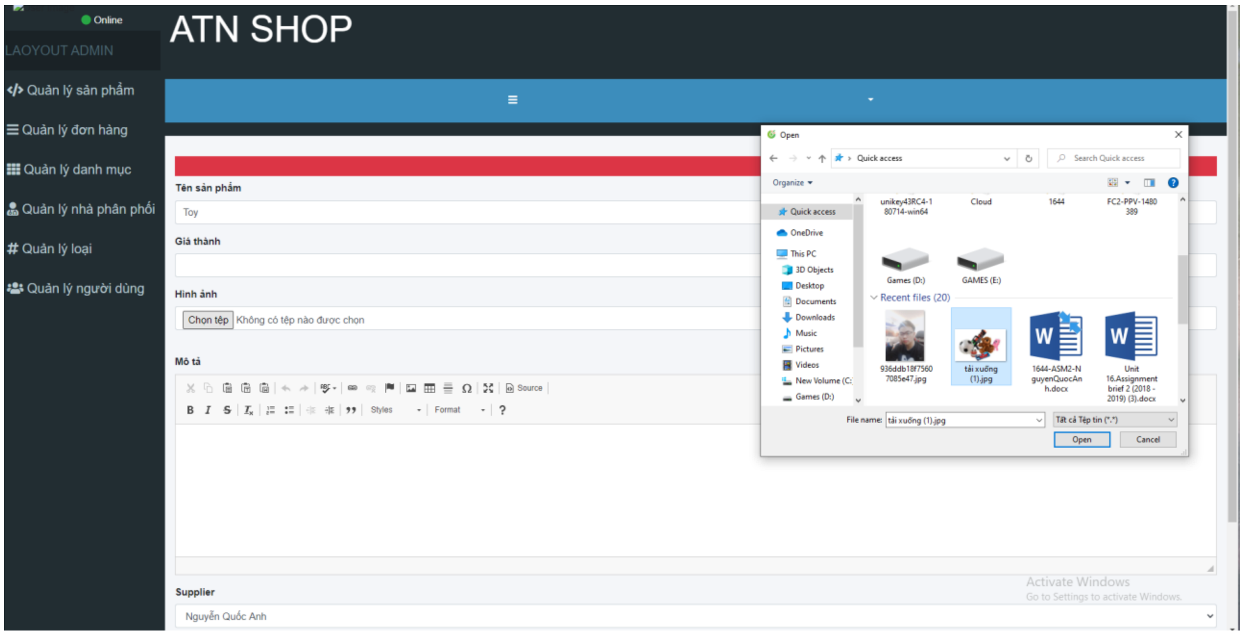
Danh sach san pham

ID	Name	Price	Description	Avatar	
5fe588f75d8da31ce8abc545	Hentai1	312312	31231312		
5fe589495d8da31ce8abc546	Hentai3123	131231	3123123		
5fe589775d8da31ce8abc547	312	312312	3123123123		

Figure 3: Item List

This is the screen showing the list of products. For each interface to show list. We will have a table to show all the attribute of item. In the last column, we will 3 function. Icon Eye is to view specific 1 product .. Icon Pen for update function. The trash icon is delete function. In addition, we also have separate interfaces for categories and suppliers.

3.3 Add Product/Category/Supplier



Tên sản phẩm: Toy

Giá thành:

Hình ảnh: Chọn tệp Không có tệp nào được chọn

Mô tả:

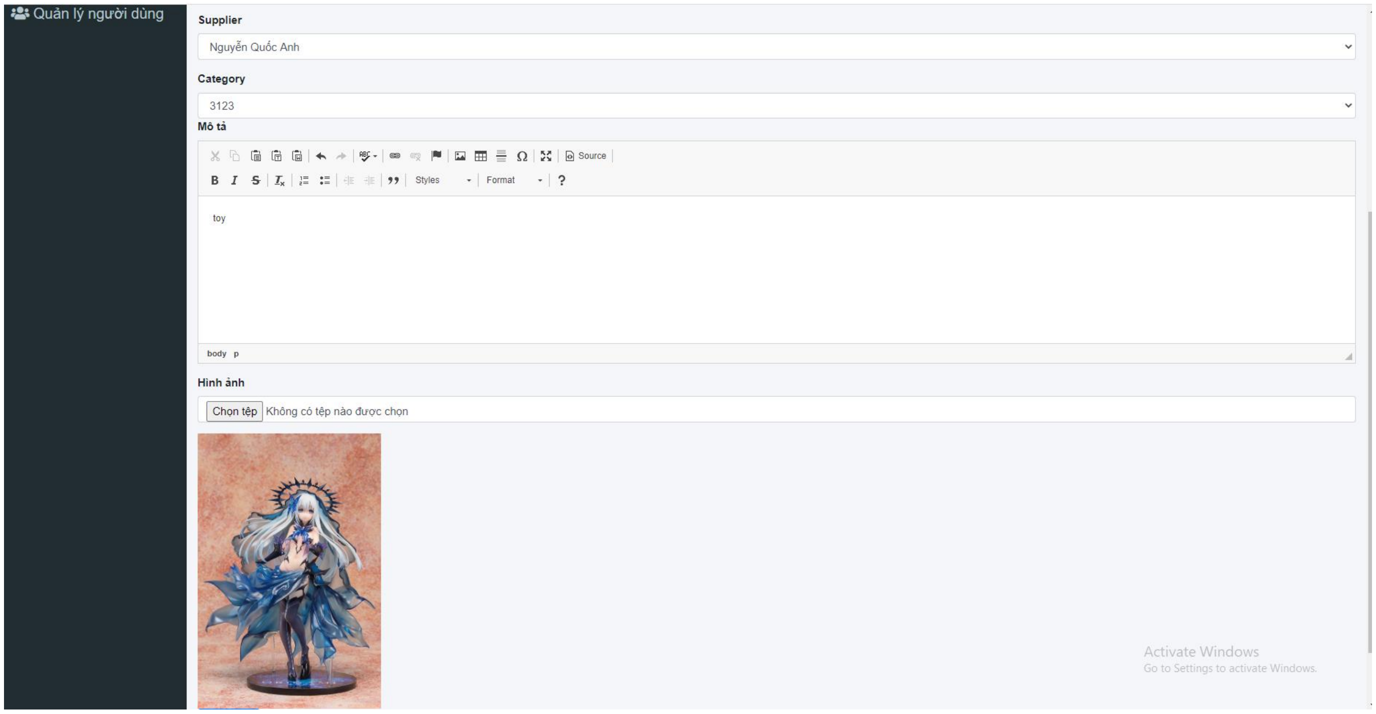
Supplier: Nguyễn Quốc Anh

File name: tải xuống (1).jpg

Tất cả Tệp tin (*.*)

Figure 4: Add Product

3.4 Update Product/Category/Supplier



Supplier
Nguyễn Quốc Anh

Category
3123

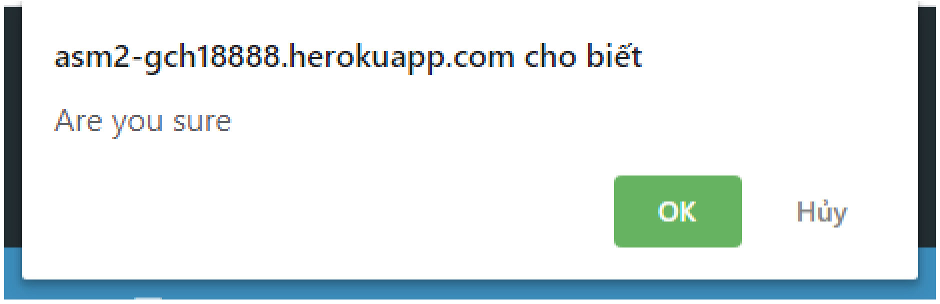
Mô tả
toy

Hình ảnh
Không có tệp nào được chọn

Figure 5: Update Product

This view will appear when you click on the pen icon in the product list. At this interface, product information will be displayed and users can update to suit their needs. This functionality is available to both categories and suppliers.

3.5 Delete Product/Category/Supplier


asm2-gch18888.herokuapp.com cho biết

Are you sure

Figure 6: Delete Product

This view will appear when clicking on the trash icon in the product list. If the user selects OK, the product will be deleted from the system. This functionality is available to both categories and suppliers.

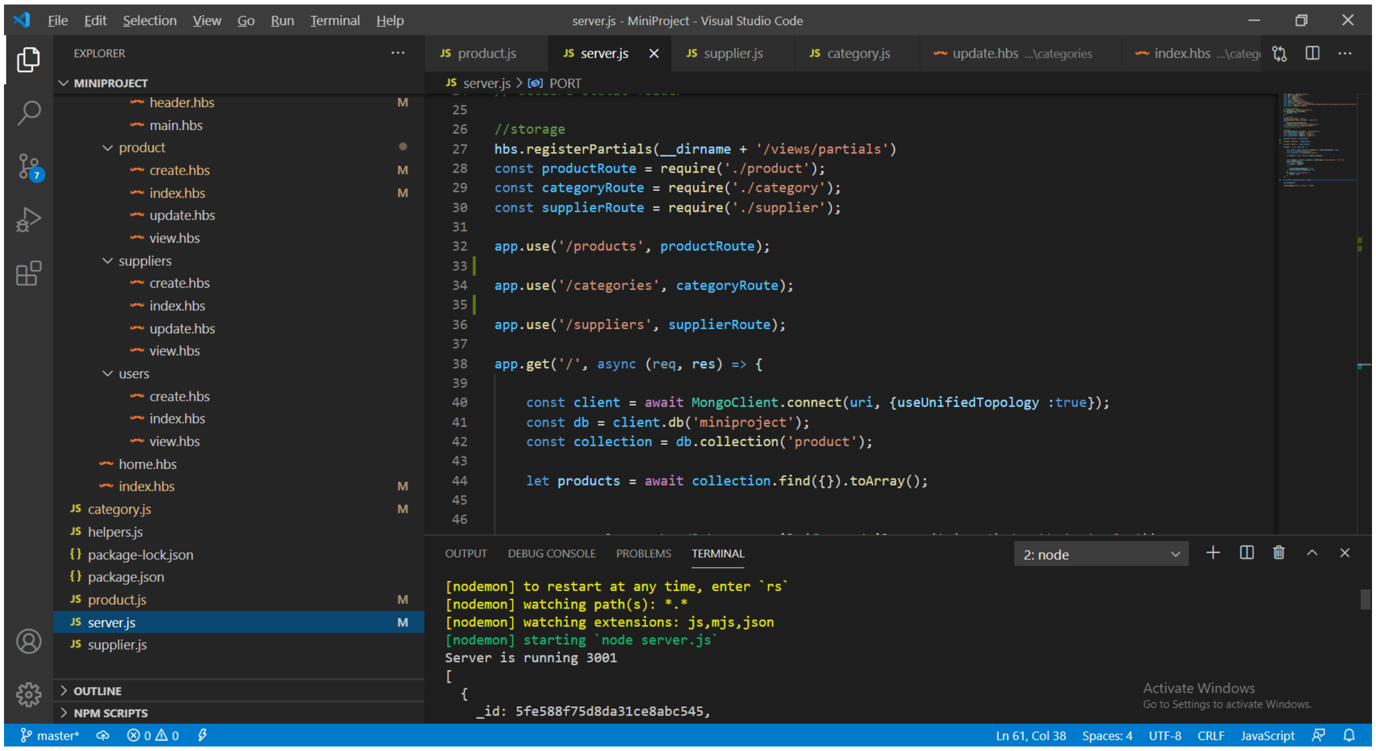
II. Code implement and deploy process

1. Tools and framework

1.1 IDE

I use Visual Studio Code because:

- It is very light in weight and takes very little time to start.
- Support many other programming different include a lot of extensions, open source
- Many shortcuts are available to make the job easy.
- Visual Studio Code support terminal command for Heroku CLI and Git CLI.



```
//storage
hbs.registerPartials(__dirname + '/views/partials')
const productRoute = require('./product');
const categoryRoute = require('./category');
const supplierRoute = require('./supplier');

app.use('/products', productRoute);

app.use('/categories', categoryRoute);

app.use('/suppliers', supplierRoute);

app.get('/', async (req, res) => {

    const client = await MongoClient.connect(uri, {useUnifiedTopology :true});
    const db = client.db('miniproject');
    const collection = db.collection('product');

    let products = await collection.find({}).toArray();

```

```
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js`
Server is running 3001
[
  {
    _id: 5fe588f75d8da31ce8abc545,
```

Figure 7: Visual Studio Code

1.2 Framework

In this project, I will use express framework. Express is a minimal and flexible Node.js web application framework that provides a powerful feature set for web and mobile applications. With a multitude of HTTP utility methods and middleware creation at your disposal, creating a powerful API is quick and easy. Express provides a rich class of basic web application features, without obscuring the Node.js features you know and love.

1.3 Source Code Manager

Github will help me manage and share source with other members of the project team. It also helps in storing and also backing up source code.

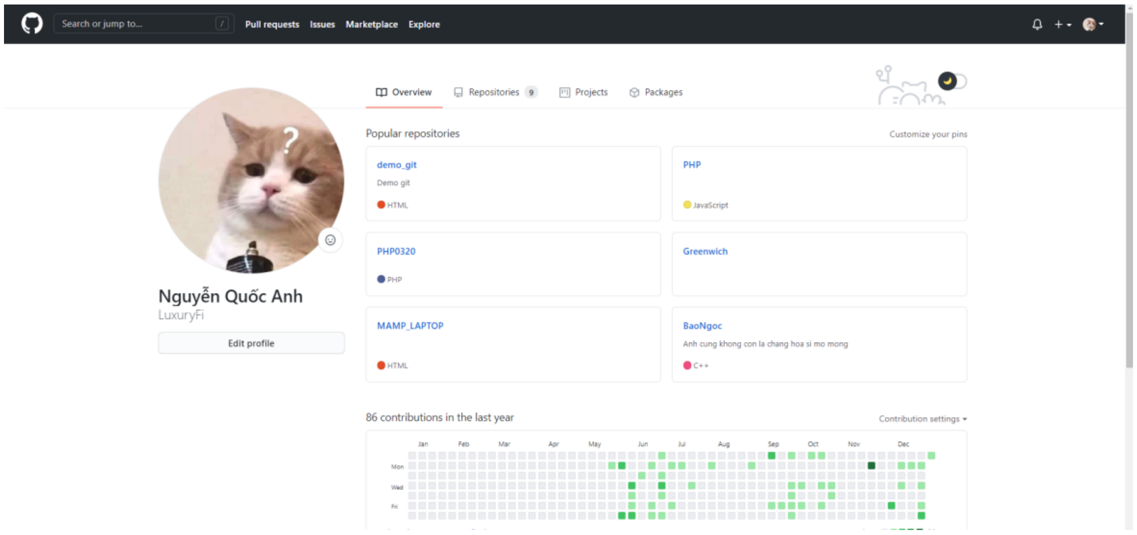
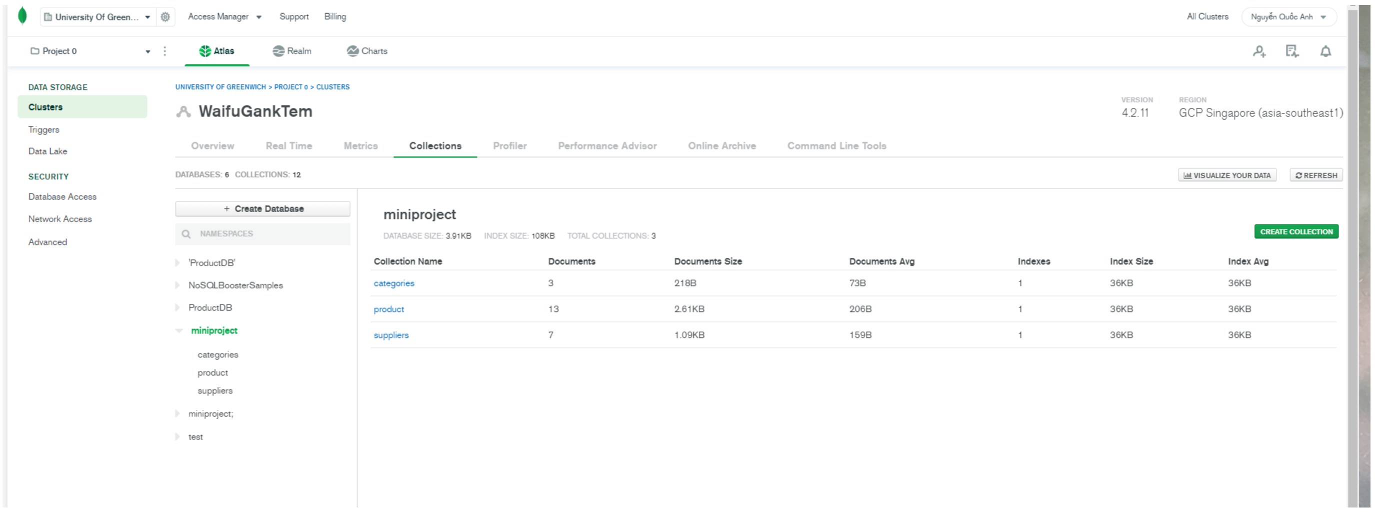


Figure 8: Github

1.4 Database server

In this project, I will use mongodb as my database because:

- It's support online database, which you need only are create a mongodb account and mongo atlas then get connection link.
- Document-oriented storage - Data is stored as JSON-style documents.
- High security.
- It's nosql database



DATABASES: 6 COLLECTIONS: 12

miniproject

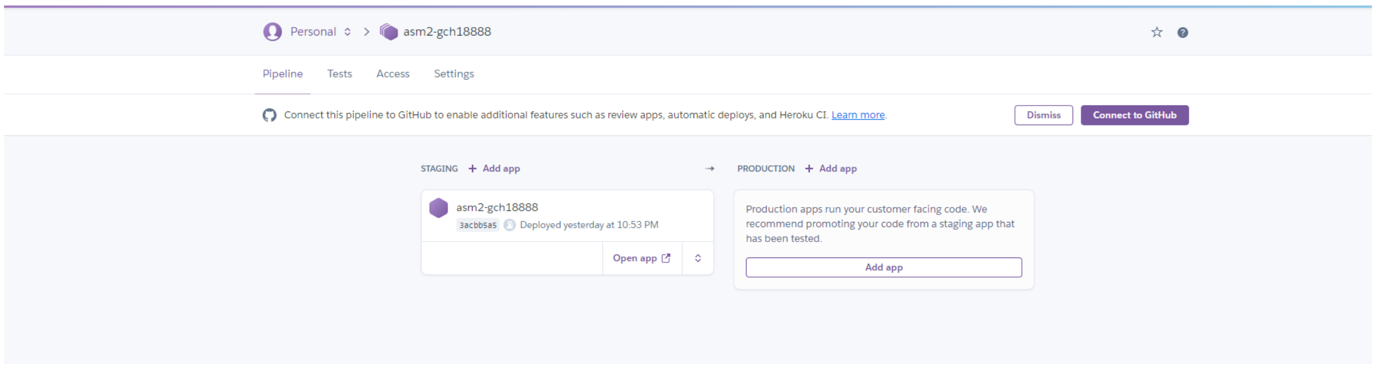
DATABASE SIZE: 3.91KB INDEX SIZE: 108KB TOTAL COLLECTIONS: 3

Collection Name	Documents	Documents Size	Documents Avg	Indexes	Index Size	Index Avg
categories	3	218B	73B	1	36KB	36KB
product	13	2.61KB	206B	1	36KB	36KB
suppliers	7	1.09KB	159B	1	36KB	36KB

Figure 9: MongoDB Atlas

1.5 Cloud computing module

In the last stage, i will use Heroku to deploy the project in internet. Heroku is cloud computing server which support us deploy the project in the PAAS model. It has the high scale and suitable for small to big company. Heroku is committed to a wide range of security policies with a professional technical team working 24/7.



Connect this pipeline to GitHub to enable additional features such as review apps, automatic deploys, and Heroku CI. Learn more.

asm2-gch18888
3acbb5a5 Deployed yesterday at 10:53 PM

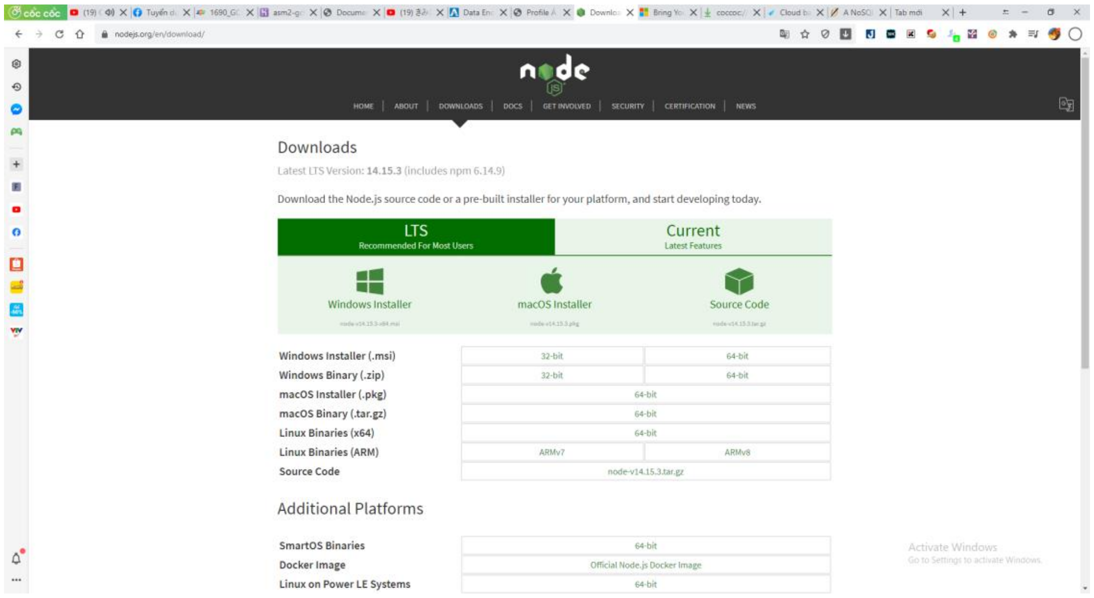
Production apps run your customer facing code. We recommend promoting your code from a staging app that has been tested.

Figure 10: Heroku

2. Deploy

2.1 Config framework express on the env

Express is a framework of nodejs, therefore the first thing which we need to do is download and install nodejs.



Downloads

Latest LTS Version: **14.15.3** (includes npm 6.14.9)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

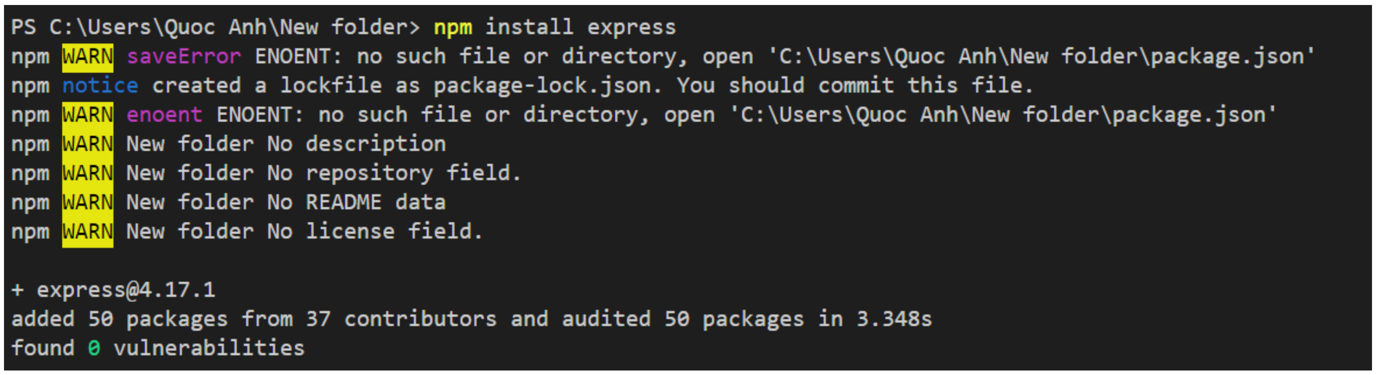
Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit	
macOS Binary (.tar.gz)	64-bit	
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v14.15.3.tar.gz	

Additional Platforms

SmartOS Binaries	64-bit
Docker Image	Official Node.js Docker Image
Linux on Power LE Systems	64-bit

Figure 11: Download Nodejs

After finish the first stage, we need to create project and user terminal command to install express with Node package manager: npm install express.



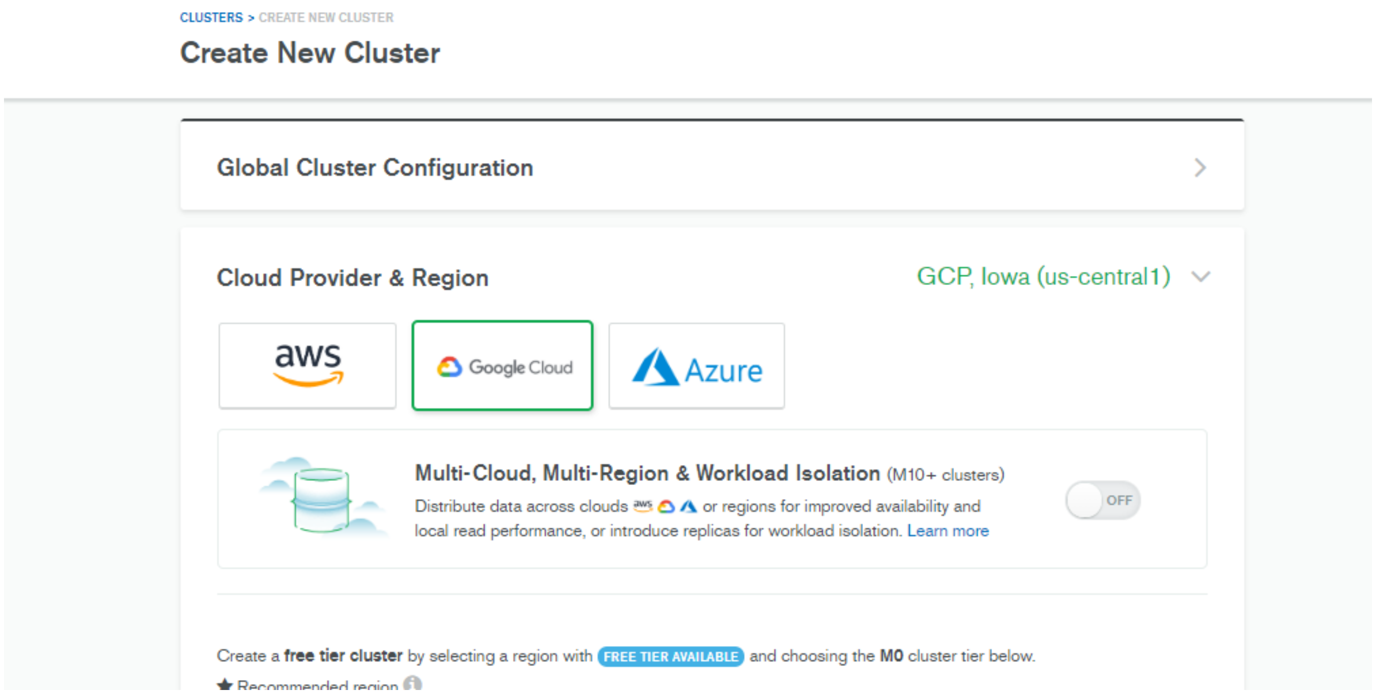
```
PS C:\Users\Quoc Anh\New folder> npm install express
npm WARN saveError ENOENT: no such file or directory, open 'C:\Users\Quoc Anh\New folder\package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN enoent ENOENT: no such file or directory, open 'C:\Users\Quoc Anh\New folder\package.json'
npm WARN New folder No description
npm WARN New folder No repository field.
npm WARN New folder No README data
npm WARN New folder No license field.

+ express@4.17.1
added 50 packages from 37 contributors and audited 50 packages in 3.348s
found 0 vulnerabilities
```

Figure 12: Install Express

2.2 Config and connect with mongodb

The next stage is connect to database. In this project, we will use mongodb atlast because it's more suitable for the project than setting up mongodb in local. Therefore we need to create mongodb atlas account.

Step 1: Create a new cluster.



Multi-Cloud, Multi-Region & Workload Isolation (M10+ clusters)

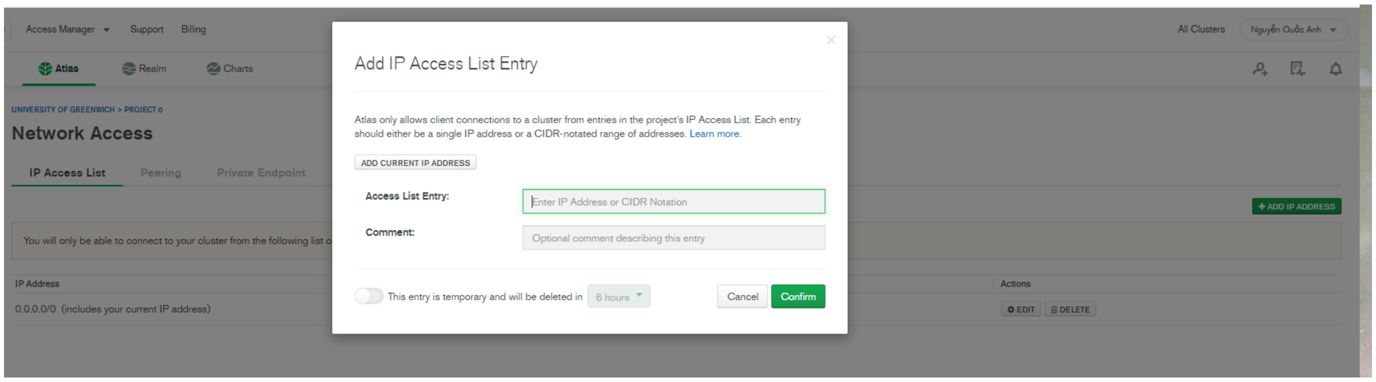
Distribute data across clouds or regions for improved availability and local read performance, or introduce replicas for workload isolation. Learn more

Create a **free tier cluster** by selecting a region with FREE TIER AVAILABLE and choosing the **M0** cluster tier below.

★ Recommended region

Figure 13: Create New Cluster in MongoDB Atlas

Step 2: In the recent cluster, click to the Network access and set the new privacy for the connect ip address.



Add IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. Learn more.

Access List Entry: Enter IP Address or CIDR Notation

Comment: Optional comment describing this entry

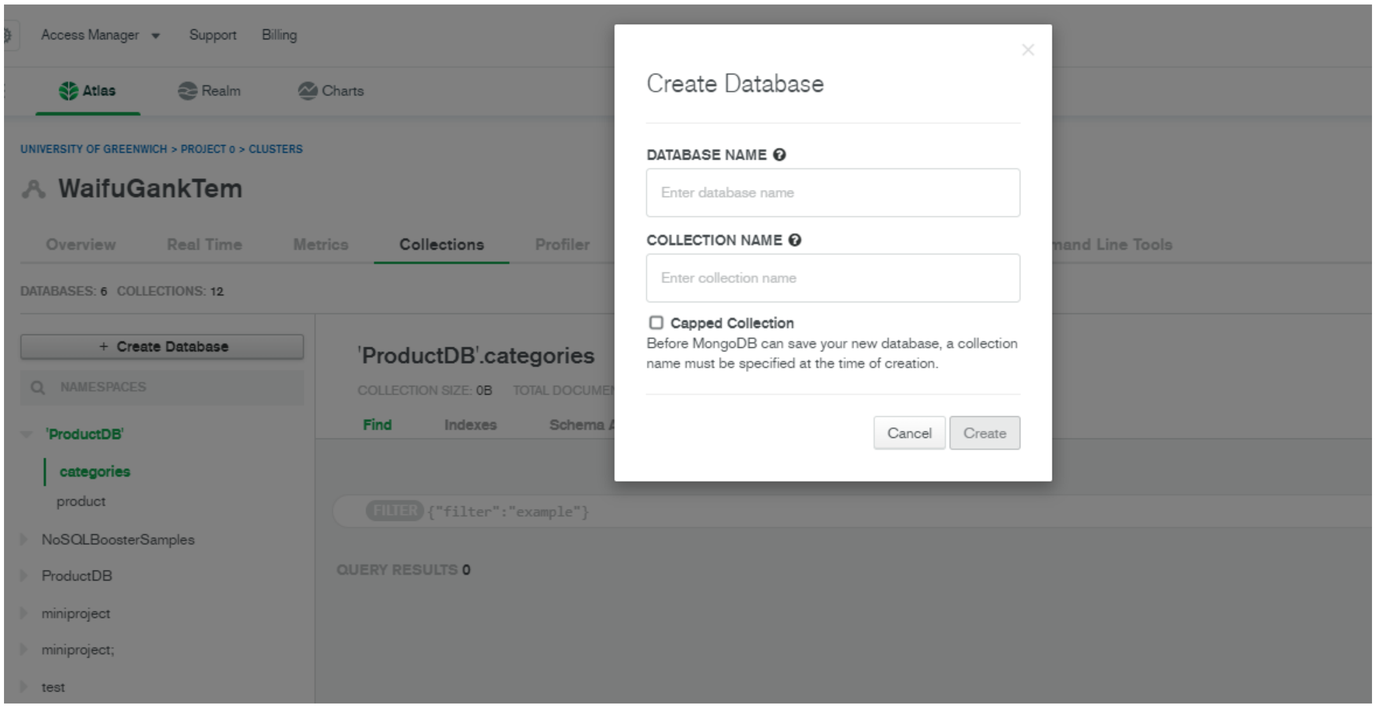
This entry is temporary and will be deleted in 6 hours

You will only be able to connect to your cluster from the following list o

IP Address	Actions
0.0.0.0/0 (includes your current IP address)	EDIT DELETE

Figure 14: Add IP Access List

Step 3: Back to cluster tab and click to the collection. Click to the create database.


Access Manager Support Billing
Atlas Realm Charts
UNIVERSITY OF GREENWICH > PROJECT 0 > CLUSTERS
WaifuGankTem
Overview Real Time Metrics Collections Profiler ...mand Line Tools
DATABASES: 6 COLLECTIONS: 12
+ Create Database
NAMESPACES
'ProductDB'
categories
product
NoSQLBoosterSamples
ProductDB
miniproject
miniproject;
test
'ProductDB'.categories
COLLECTION SIZE: 0B TOTAL DOCUME...
Find Indexes Schema...
FILTER {"filter":"example"}
QUERY RESULTS 0

Create Database

DATABASE NAME
Enter database name

COLLECTION NAME
Enter collection name

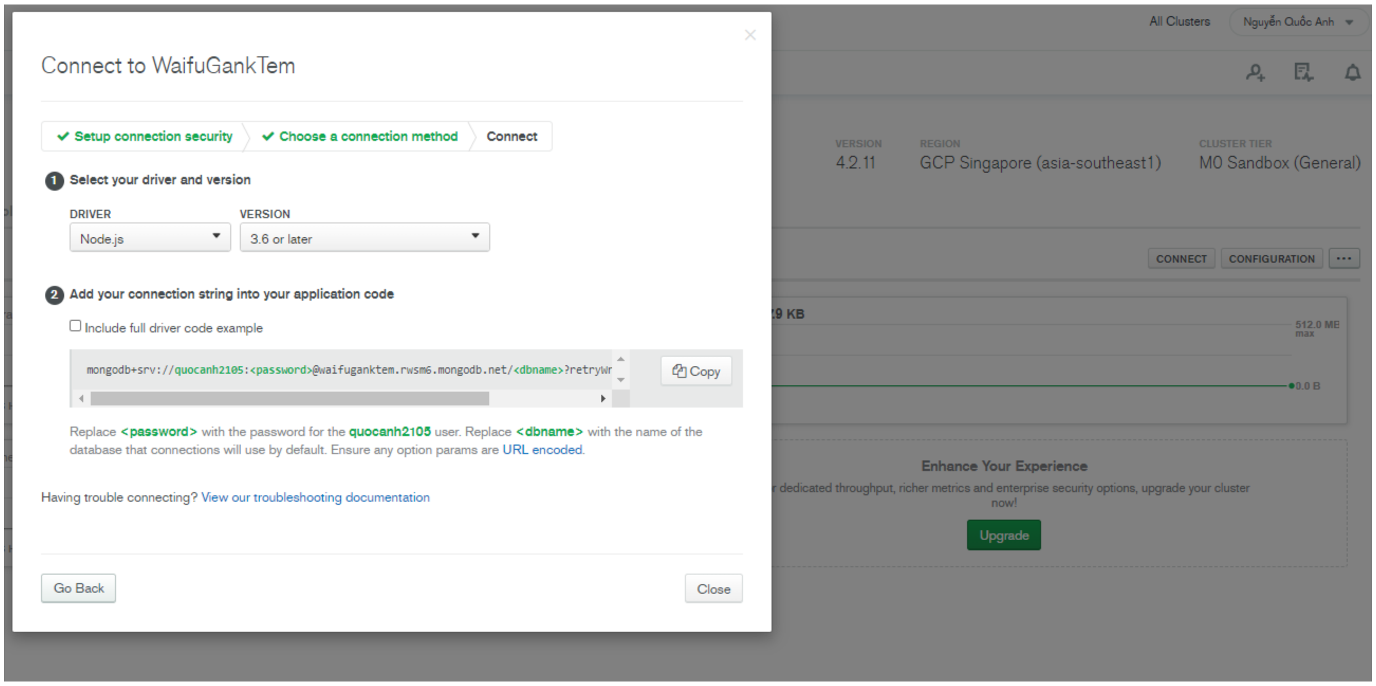
Capped Collection

Before MongoDB can save your new database, a collection name must be specified at the time of creation.

Cancel Create

Figure 15: Create Database

Step 4: Choose the created cluster and click connect to get the connection string. You need to replace the password with you current password


All Clusters Nguyễn Quốc Anh
VERSION 4.2.11 REGION GCP Singapore (asia-southeast1) CLUSTER TIER M0 Sandbox (General)
CONNECT CONFIGURATION ...

Connect to WaifuGankTem

✓ Setup connection security ✓ Choose a connection method Connect

1 Select your driver and version

DRIVER: Node.js
VERSION: 3.6 or later

2 Add your connection string into your application code

Include full driver code example

```
mongodb+srv://quocanh2105:<password>@waifuganktem.rwsm6.mongodb.net/<dbname>?retryWr
```

Copy

Replace <password> with the password for the quocanh2105 user. Replace <dbname> with the name of the database that connections will use by default. Ensure any option params are URL encoded.

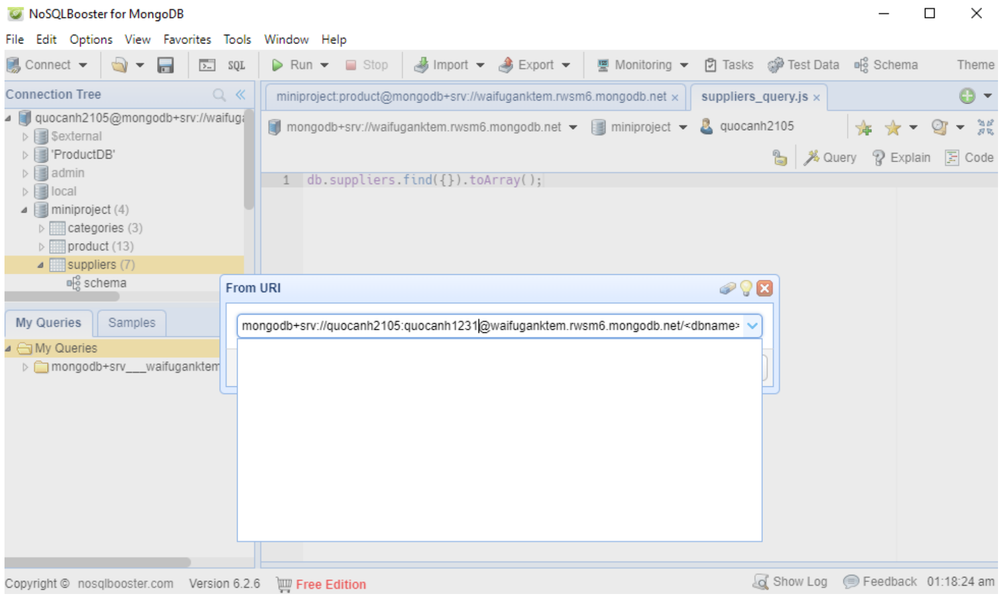
Having trouble connecting? View our troubleshooting documentation

Go Back Close

Enhance Your Experience
...r dedicated throughput, richer metrics and enterprise security options, upgrade your cluster now!
Upgrade

Figure 16: Get Connection String

Step 5: Connect to database with noSQLBooster. Click to File->Quick Connect and paste the connection string to url.


NoSQLBooster for MongoDB
File Edit Options View Favorites Tools Window Help
Connect SQL Run Stop Import Export Monitoring Tasks Test Data Schema Theme
Connection Tree
quocanh2105@mongodb+srv://waifug
$external
'ProductDB'
admin
local
miniproject (4)
categories (3)
product (13)
suppliers (7)
schema
My Queries Samples
My Queries
mongodb+srv___waifuganktem
miniproject:product@mongodb+srv://waifuganktem.rwsm6.mongodb.net
suppliers_query.js
mongodb+srv://waifuganktem.rwsm6.mongodb.net
miniproject
quocanh2105
Query Explain Code
From URI
Copyright © nosqlbooster.com Version 6.2.6 Free Edition
Show Log Feedback 01:18:24 am

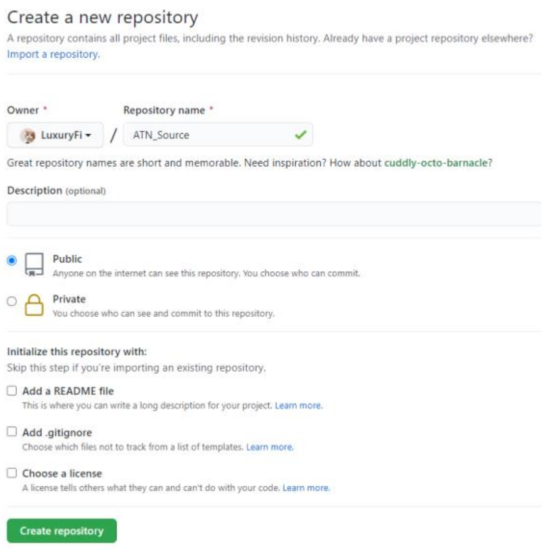
```
db.suppliers.find({}).toArray();
```

```
mongodb+srv://quocanh2105:quocanh1231@waifuganktem.rwsm6.mongodb.net/<dbname>
```

Figure 17: NoSQLBooster connect with mongodb

2.3 Config git and upload file to github

During web application development, I need to use Github to manage source and deploy it in heroku to test. For each project, I will create one repository to manage.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * Repository name *

LuxuryFi / ATN_Source ✓

Great repository names are short and memorable. Need inspiration? How about cuddly-octo-barnacle?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ Add a README file
This is where you can write a long description for your project. Learn more.

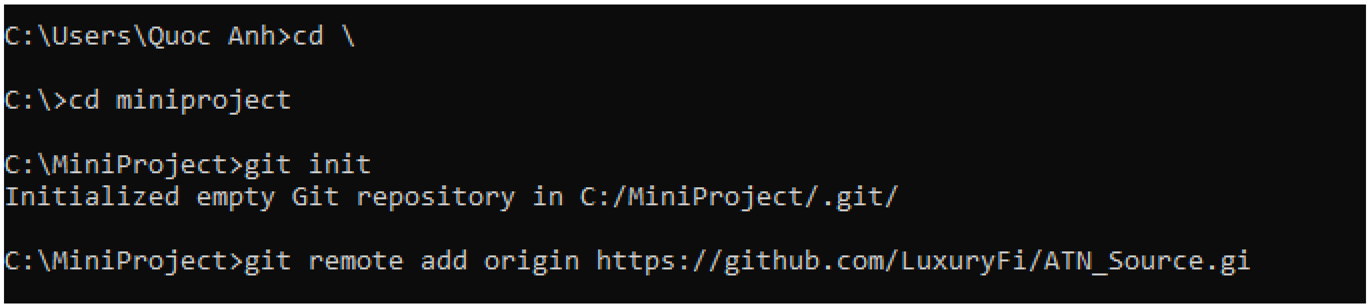
☐ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

☐ Choose a license
A license tells others what they can and can't do with your code. Learn more.

Create repository

Figure 18: Create new repository

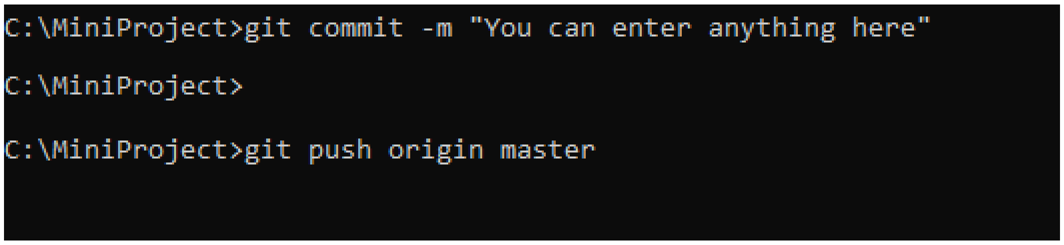
After created a repository, I will use command to sync the current project with git



```
C:\Users\Quoc Anh>cd \

C:\>cd miniproject

C:\MiniProject>git init
Initialized empty Git repository in C:/MiniProject/.git/

C:\MiniProject>git remote add origin https://github.com/LuxuryFi/ATN_Source.gi
```

Figure 19: Sync folder with repository

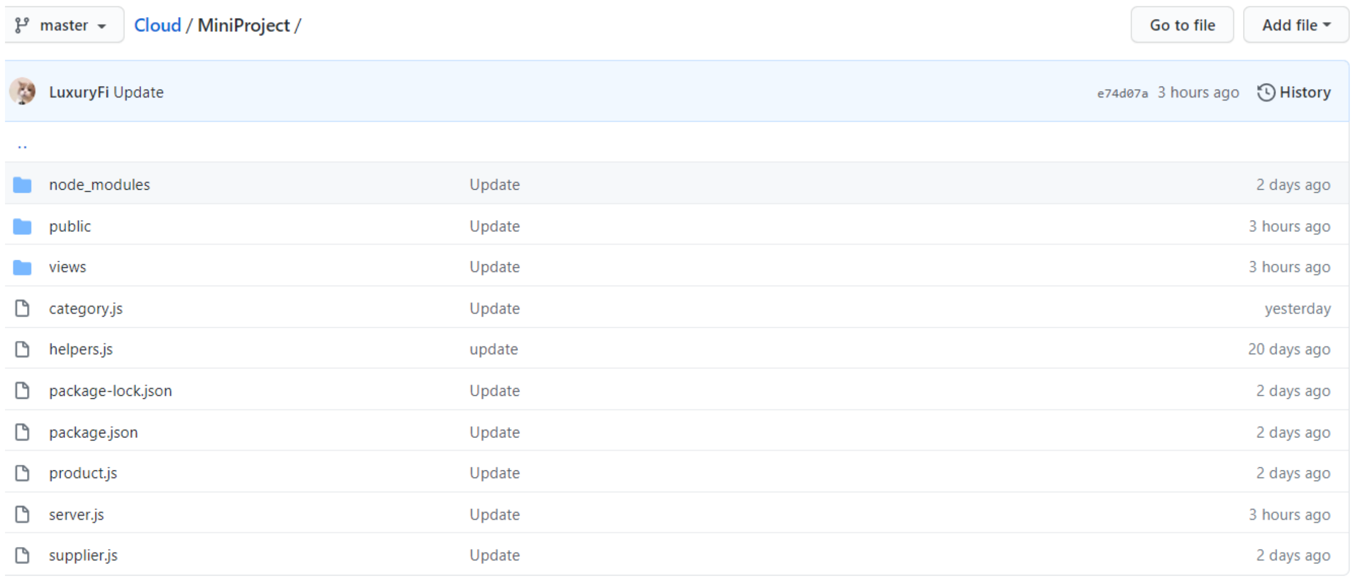
Then each time I want to sync my code, I can use some command are:



```
C:\MiniProject>git commit -m "You can enter anything here"

C:\MiniProject>

C:\MiniProject>git push origin master
```

Figure 20: Push code

You can find the source code in your repository in github if upload successful.



master / Cloud / MiniProject /

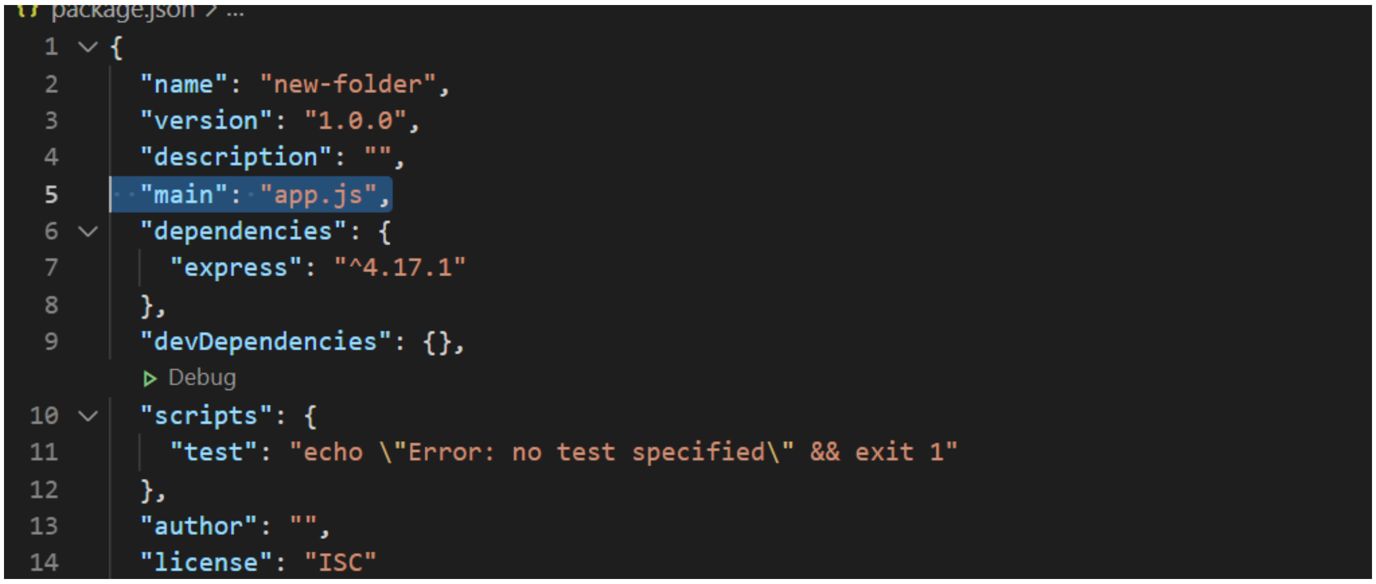
LuxuryFi Update — e74d07a 3 hours ago — History

..		
node_modules	Update	2 days ago
public	Update	3 hours ago
views	Update	3 hours ago
category.js	Update	yesterday
helpers.js	update	20 days ago
package-lock.json	Update	2 days ago
package.json	Update	2 days ago
product.js	Update	2 days ago
server.js	Update	3 hours ago
supplier.js	Update	2 days ago

Figure 21: Git repository after push

2.4 Deploy code on Heroku server

To be able to deploy the project to heroku, we need to configure something before. In the visual studio code terminal command. We will create package.json with npm init command.

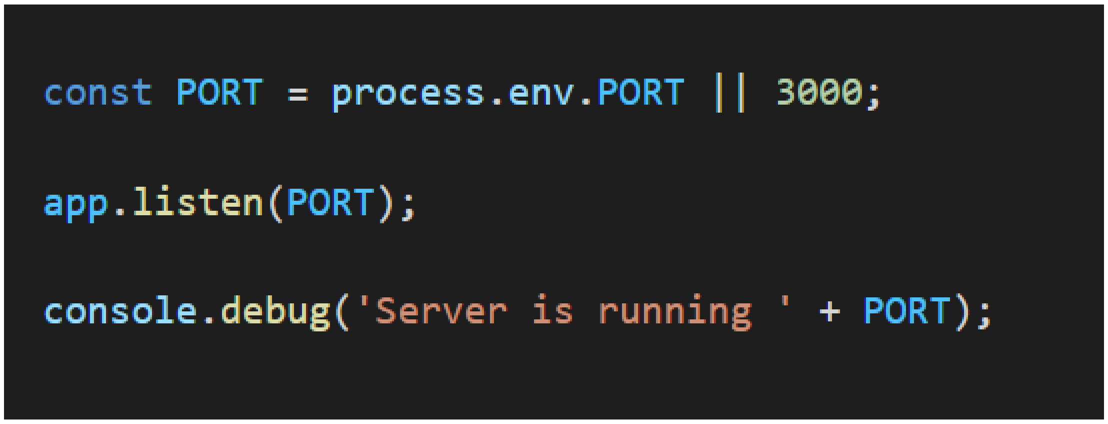


```
{} package.json > ...
{
  "name": "new-folder",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "dependencies": {
    "express": "^4.17.1"
  },
  "devDependencies": {},
  ▷ Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
```

Figure 22: Package-Json

Note that you should use the npm init command while opening the code file that is the root / main file of your project. or you need to change it to your root / main file in package.json file.

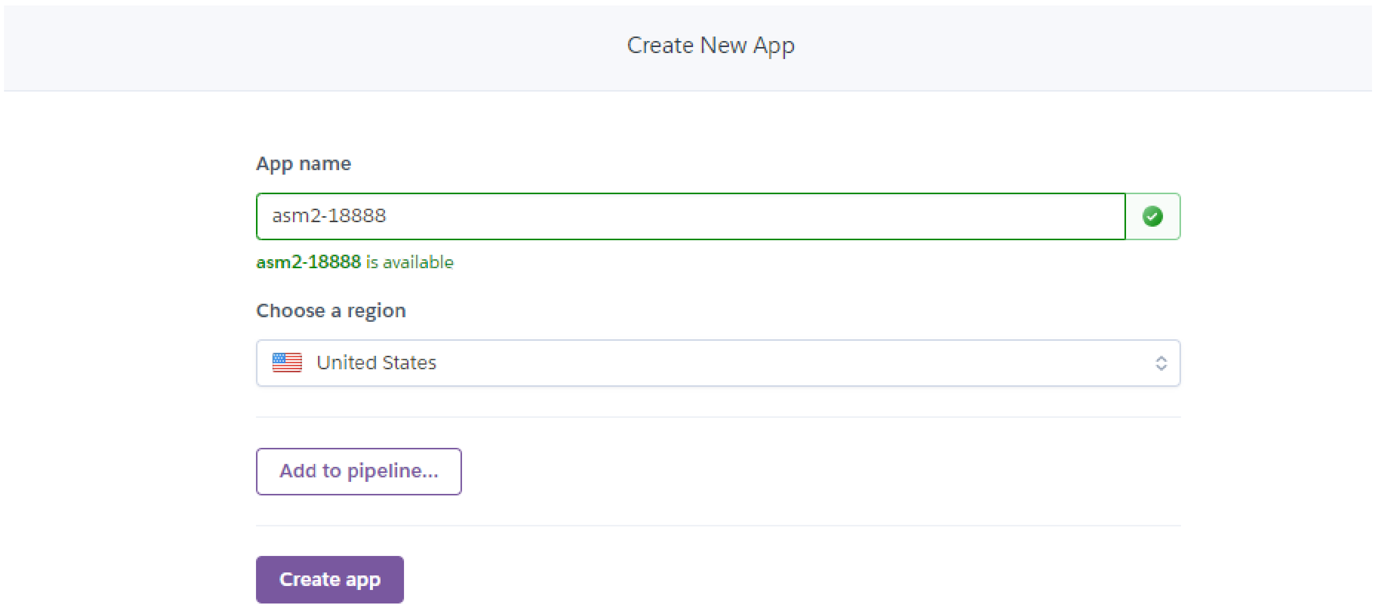
Beside, you web application will be public in the internet so we need to change that PORT to process.env.PORT to app.listen.



```
const PORT = process.env.PORT || 3000;

app.listen(PORT);

console.debug('Server is running ' + PORT);
```

Figure 23: Setup app.listen and PORT

In the next stage, i will register a new account and click to the create new app button in heroku homepage.


Create New App

App name

asm2-18888

asm2-18888 is available

Choose a region

United States

Figure 24: Create Herroku Web Application

Because my web application is written with nodejs. So we need to add Nodejs buildpack for our app in heroku.

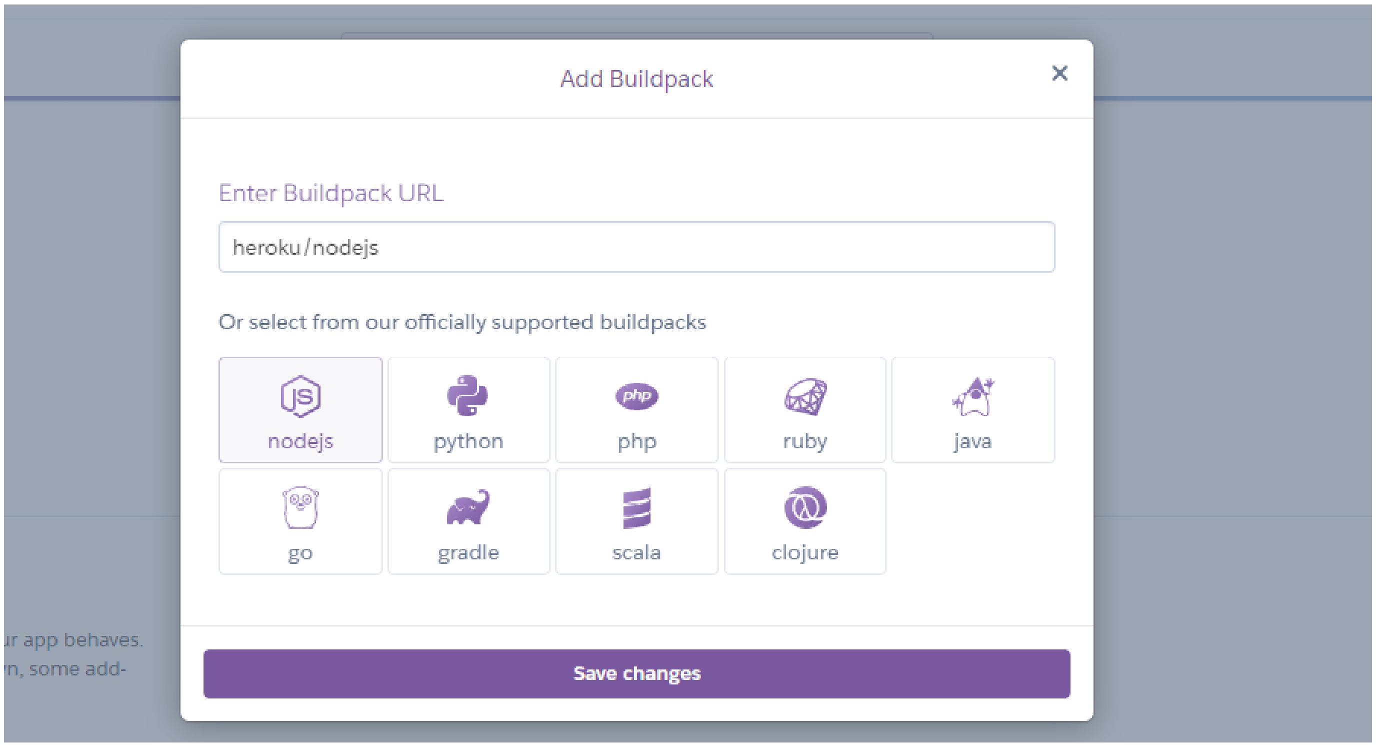
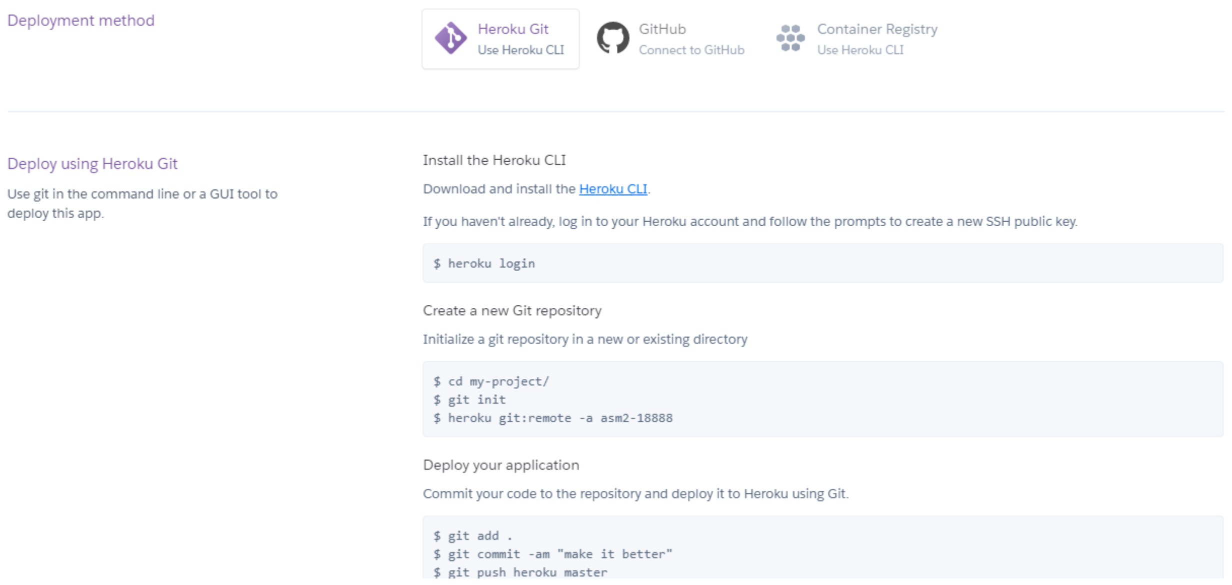


Figure 25: Add BuildPack nodejs

In last stage, we have 2 main ways to deploy our project in heroku

- Choose connect to github and connect your repository.
- Use Heroku CLI (you need to install it before). Heroku also provide some guide to use heroku CLI for deploy your project.



Deploy using Heroku Git

Use git in the command line or a GUI tool to deploy this app.

Install the Heroku CLI

Download and install the Heroku CLI.

If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Create a new Git repository

Initialize a git repository in a new or existing directory

```
$ cd my-project/
$ git init
$ heroku git:remote -a asm2-18888
```

Deploy your application

Commit your code to the repository and deploy it to Heroku using Git.

```
$ git add .
$ git commit -am "make it better"
$ git push heroku master
```

Figure 26: Documentation for using heroku

After push or connect successful to your project. You can click "open app" and see your project.

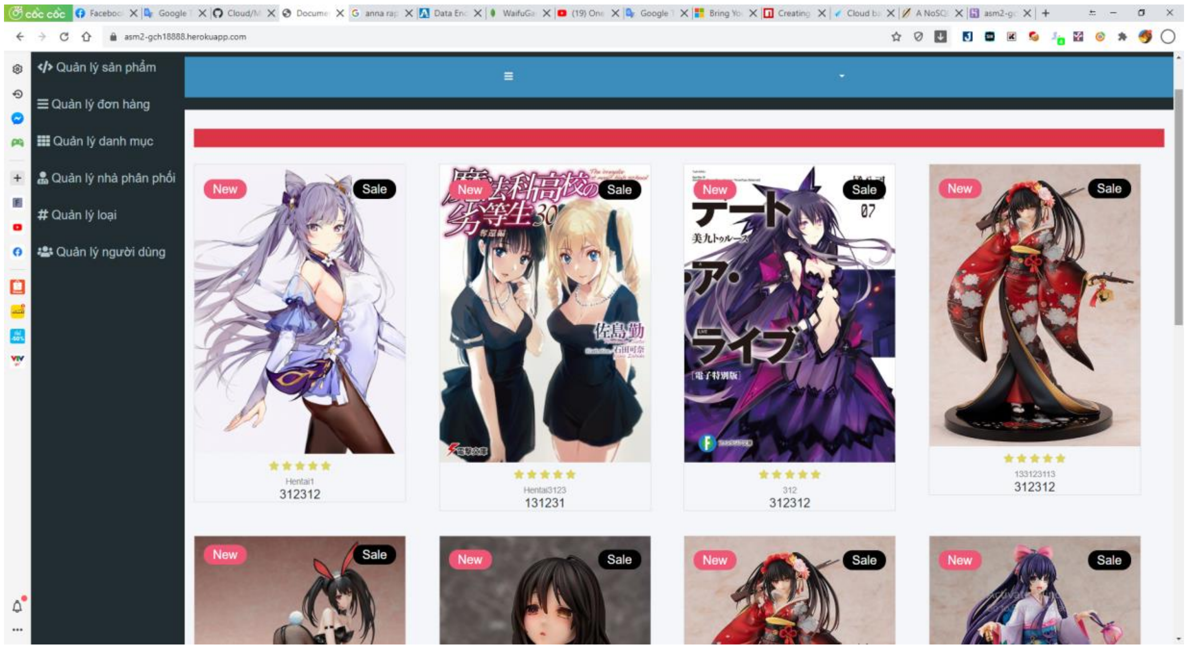


Figure 27: Heroku Web In Internet

3. Code implement

4. Source code and website

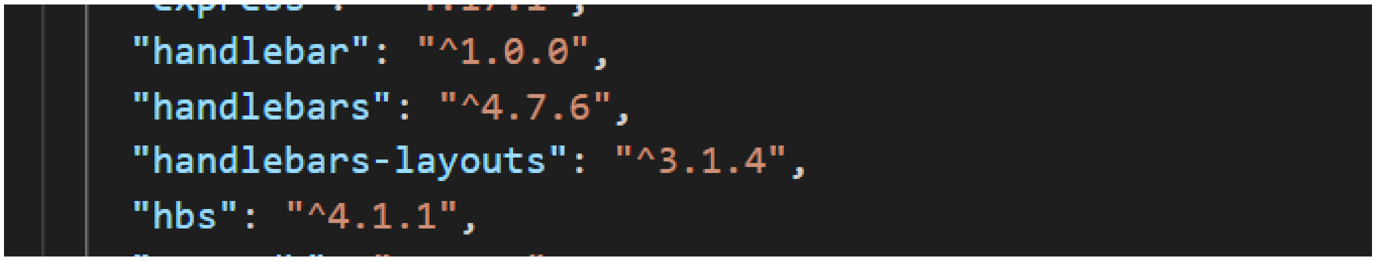
III. Difficulties which one can face during the development process.

There are many difficulties when i develop and implement this project. But i did research and learn how to fix it. In this part, i will list some difficulties which i have to encountered and how i can overcome them.

1. Programming

a) Difficulties

Other than that this is my first time using it, I made quite a few mistakes during development. Nodejs has too many different frameworks like Express, NestJs, SailJs, KoaJs. Each framework has some different pros and cons. This makes me very confused in choosing the right framework for the project. After choosing a suitable framework, we have to choose the suitable view engine ... Because Nodejs is an open source platform, so it has the advantage of abundant community-resources not like other closed platform like asp, javaspring, ... This leads to inconsistency in naming libraries. These libraries all have similarities and differences, so it can be difficult to distinguish when searching for tutorial articles. During the first use, when i dont remember the exactly name of handlebars and only install handlebar library but it's still work till i have some bugs and need to uninstall and reinstall handlebars. Each library has its own specific syntax, so I confused a lot when looking up their functions.

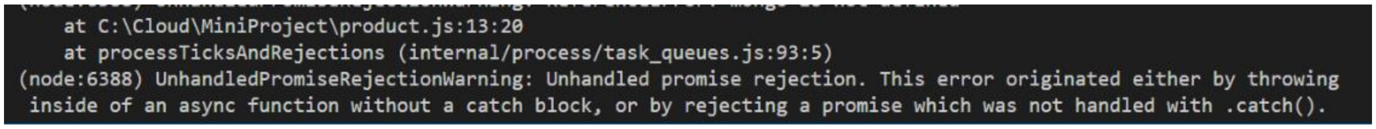


```
"handlebar": "^1.0.0",
"handlebars": "^4.7.6",
"handlebars-layouts": "^3.1.4",
"hbs": "^4.1.1",
```

Figure 28: Nodejs library

Next, I ran into a problem working with the view engine. Different from writing server side and client-side code together in one file. Source Code is structured in a model that is almost equivalent to MVC. Hence transferring the data between my view and routes in the early stages encountered some difficulties.

Beside, when working with data, I had many errors about asynchronous processing. Unlike some other server-side languages, javascript strictly handles asynchronous handling. It is easy to make an error if we use the inappropriate callback functions.



```
    at C:\Cloud\MiniProject\product.js:13:20
    at processTicksAndRejections (internal/process/task_queues.js:93:5)
(node:6388) UnhandledPromiseRejectionWarning: Unhandled promise rejection. This error originated either by throwing
 inside of an async function without a catch block, or by rejecting a promise which was not handled with .catch().
```

Figure 29: Asynchronous Bug

b) Solution

Faced with the above problems, the first thing I did was to thoroughly re-learn how nodejs works. By understanding the basic keywords as well as operating principles, I analyzed pros, cons and chose express as framework for the project. Then I chose hbs view engine because its syntax and keywords are easy to remember. In addition, hbs has a lot of users, so it's easy to look up information. because nodejs is inconsistency in naming libraries. so I'll just read the articles and documentation on the nodejs homepage. Also, when consulting the tutorials on forums, I need to look up exactly which library I am using. For example, with handlebars viewengine I have 2 libraries: hbs and handlebars. They have a lot of similarities such as supporting 1 view engine, compile, register helper, ... However, I will only select 1 library for the whole article to ensure consistency.

For asynchronous handling I have chosen to use async function and await instead of the callback function. Async / await is a special syntax that makes working with Promises easier. When using async / await, the structure of an asynchronous handler will be more similar to a synchronous handler. If handled this way, even if I make more requests, the program structure is still very clear and coherent.

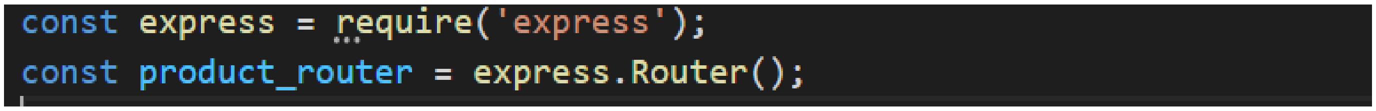
2. Debugging

a) Difficulties

when i start to write code, all code are written in the one file is server.js. it make debug become more hard but easier to get error. When the software is too functional, route naming will easily be duplicated. When a bug occurs, it's very hard to find the module which is related to the bug. In addition, passing data between the views and routes also sometimes leads to errors and makes it difficult to detect the cause. Because, the cause may be database or form.

b) Solution

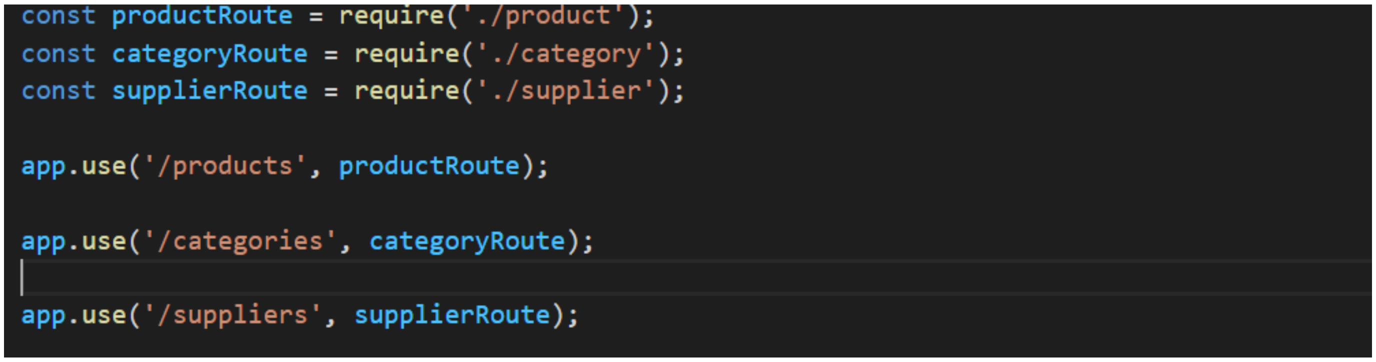
To resolve the problem, i have use router function of express. Which router function, i can divide the app.js to many other route with different features. The names of the routes in each router may overlap. It makes finding bugs easier, when a function occur errors, i can find exactly which router and route relate to it.



```
const express = require('express');
const product_router = express.Router();
```

Figure 30: Create Router In Express

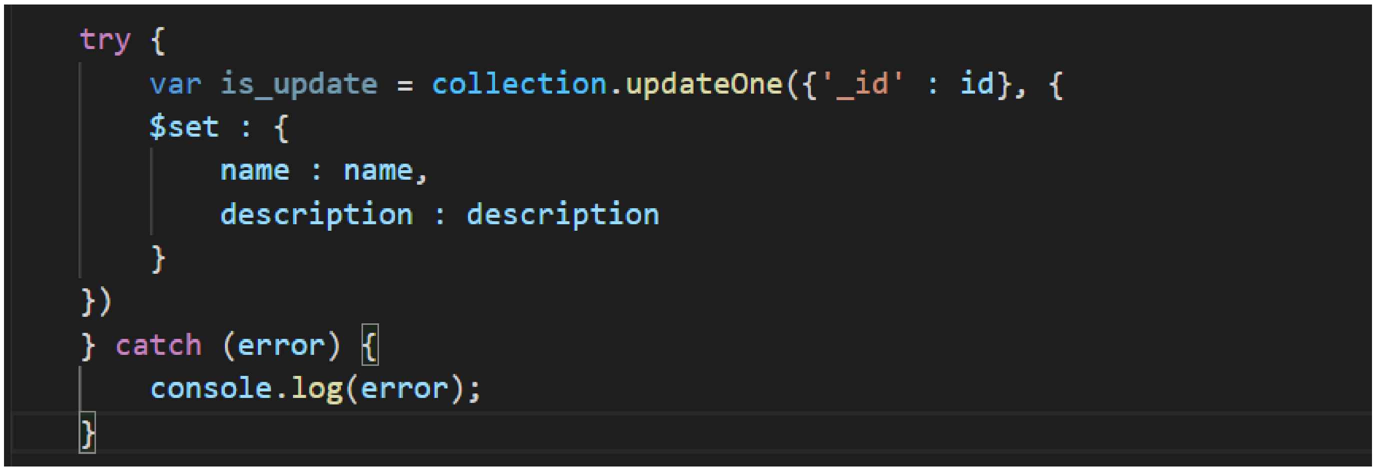
In figure below, I have divided app to 3 main router. Product router has all the function relate to the product like update, delete , add … It’s the same for category and supplier router.



```
const productRoute = require('./product');
const categoryRoute = require('./category');
const supplierRoute = require('./supplier');

app.use('/products', productRoute);

app.use('/categories', categoryRoute);

app.use('/suppliers', supplierRoute);
```

Figure 31: Embed Router In Main file

Beside, i have to use try catch and console.log in debugging. It helps me catch all the possible exeception. I can also check the data if it can send from form.

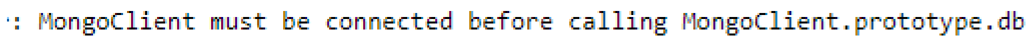


```
try {
    var is_update = collection.updateOne({'_id' : id}, {
    $set : {
        name : name,
        description : description
    }
})
} catch (error) {
    console.log(error);
}
```

Figure 32: Try Catch

3) Deploying

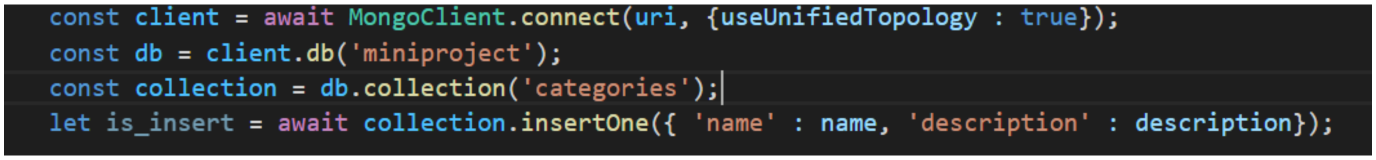
a) Database: I spent a lot of time connecting to the database. In the first time i tried to connect to database, it got a lot of errors and I don't know what it means. There is many other guide for that problem. I have tried many of them but are not successful.



```
: MongoClient must be connected before calling MongoClient.prototype.db
```

Figure 33: Database Bug

Solution: I have connected according to documentation in the mongodb homepage, in addition I use asynchronous handling in database connection to ensure stability.



```
const client = await MongoClient.connect(uri, {useUnifiedTopology : true});
const db = client.db('miniproject');
const collection = db.collection('categories');
let is_insert = await collection.insertOne({ 'name' : name, 'description' : description});
```

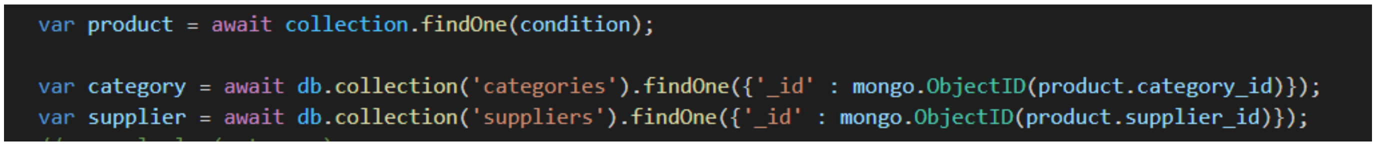
Figure 34: Connection to database code

Besides, mongodb is a nosql database, so I have had some difficulty performing nested queries from multiple tables. This is necessary when I need to extract some category and supplier information from a product.

Because, when adding a product to database, I will use the category and supplier fields as id to avoid duplication. With relational database, i can use JOIN to JOIN other collection (tables) for each document (row). There are a library which help me to do it with mongodb. But it breaks down the properties that are considered to be mongodb's advantages over other database management systems.

Solution:

When performing a query with a product, I retrieve the category and supplier id fields in the product and perform two more queries on the other 2 collections. This yields the same results as a nested query, but does not need to use any additional libraries but still retains data flexibility.



```
var product = await collection.findOne(condition);

var category = await db.collection('categories').findOne({'_id' : mongo.ObjectID(product.category_id)});
var supplier = await db.collection('suppliers').findOne({'_id' : mongo.ObjectID(product.supplier_id)});
```

b) Git: When using git, I ran into problems when the github GUI repositories did not sync with repositories on the web. Using git in the console was quite difficult for me as I did not remember all the commands.

Solution: I searched for information about that problem on forums like stackoverflow and found the cause. The Git GUI will only show the repositories in the Github directory in the document. So when I create a new repository I will create in that directory. Also I have tried to learn the command commands and it was not too difficult. Now i have switched to using command instead of GUI.

C. SECURITY

I. Some issue of Cloud computing platform

1. Public Cloud

a) Issue

Loss of control. When you outsource to the public cloud, literally out of reach. Any configuration and other aspects of IT management are left to a group that are not directly involved in daily operations Low security level, because data often shares common location, not as secure as other Cloud models (Nayyar, 2019).

b) Solution

Choose from reputable providers who have a clear policy and commitment to ensuring the security of your information and your interests.

2. Private Cloud

a) Issue

Private clouds have strict restrictions on access.. when a company's employees were working outside the office, they have some difficulties to connect to the private cloud lead to securely access any applications, data or files they needed from the company servers (Nayyar, 2019).

b) Solution

Use VPN, VPN has two main advatange in this case

- Cost-effectively provide employees, wherever they are, with a secure connection to both the company's cloud and data center-based applications and data.
- Authenticate users and ensure comprehensive, consistent security without having to purchase expensive hardware or networking equipment, or add IT complexity.

3. Hybrid Cloud

a) Issue

Hybrid cloud is the combination of public cloud and private cloud. Therefore, it has all the issue of two kind cloud. In addition, the issue of technology and optimization also needs to be carefully considered (Nayyar, 2019).

b) Solution

Careful analysis of the size and usage of the system is required to come up with an appropriate model. Depending on the size and scale of the system, we need to calculated about the security and technical aspects of the system is required in order to avoid technical errors and security mishaps.

II. Security issues and solution in cloud computing enviroment.

1. Data Loss/Leakage

Cloud-based environments make it easy to share the data stored within them. These environments are accessible directly from the public Internet and include the ability to share data easily with other parties via direct email invitations or by sharing a public link to the data (Team, 2020).

Although this is one of the main advantages of cloud, the ease of data sharing in the cloud raises significant concerns about data loss or leakage. In fact, as their main cloud security issue, 69 percent of organizations point to this. Data exchange using shared connections or setting up a public cloud-based archive makes it open to those with link information, and there are unique resources for searching for these unsecured cloud installations on the Internet (Team, 2020) .

Solution:

To ensure the security of the organization's data against the above-mentioned threats of illegal discovery and loss, the Organization should be equipped with a data security solution. Overall data, this total data security solution must meet at least the following requirements:

- Data classification and standardization requirements
- Data confidentiality requirements
- Requirements for the ability to monitor, detect and prevent unauthorized transmission of data outside of the organization
- Requirements for effective operation of the solution set

(Chandra, 2017)

2. Physical Security Risk

The security measures are useless if an attacker can connect directly to the system. That way, they'll be able to bypass the majority of security systems and third-party security services like cloudflare. Besides, criminals can directly harm the physical servers of the cloud (Aslan, 2012).

Solution: The physical location of the cloud data center must be secured by the CSP in order to prevent unauthorized on-site access of CSC data. In addition, there should be security policies such as system guards or supervisors. There are security systems such as fingerprint scanning, keycard access and biometric scans to restrict access to sensitive locations in the data center .

3. Organization

If a Cloud Service Provider (CSP) leaves business or is purchased by another entity. There may be a place The threat of the organization's malicious insiders who might harm using the data generated by their CSCs(Customer/Consumer of Cloud Service) .

Solution: The risk of malicious workers being hired by a CSP can be mitigated by Putting strict legal restrictions on contracts when recruiting employees. A thorough review of the CSP As well as a comprehensive security breach notification process. On the user side, it is important to work closely with the new distributor's representative to clarify the contractual terms. Besides, users also need to prepare forms to secure data (Hein, 2019).

4. Risks of Compliance and Audit

There are law-related threats. That is, risks associated with lack of jurisdictional information, changes in jurisdictional information, Jurisdiction, the contract's unlawful provisions and pending legal conflicts. Depending on venue, for instance, Any CSPs, if requested by the government, may be required by law to turn over confidential information (Worlanyo, (2015).

Solution: This area deals mainly with legal problems and, as such, both Cloud Service Providers (CSPs) and CSCs need to consider legal and regulatory responsibilities and ensure the CSCs (Cloud Service Customer/Consumer) That these duties be met by any contracts made. The CSP should also ensure that its capabilities for exploration are Security and data protection are not affected. Having seen several approaches used in the next four fields to avoid safety lapses, in the next one We shall look at a subsection of some of the primary techniques used for ensuring data security (Worlanyo, (2015)

5. Technology security risks

These risks are the failures associated with the hardware, technologies and services provided by the CSP (Cloud Service Provider) In the public cloud, with its multi tenancy features, these include resource sharing isolation problems, and risks related to changing CSPs, i.e. portability. Regular maintenance and audit of infrastructure by CSP is recommended (Worlanyo, (2015).

Solution: Virtualized trust protection focused on defence and integrity - CSP may use the following Structure: a DHT-based overlay network hierarchy, with particular tasks to be performed by each layer. The lowest layer deals with the accumulation of reputations and colluders being probed. The top stratum deals with Similar attacks. The aggregation of credibility here is linked to the use of multiple sources to check those connections and probing colluders refer to testing if known malignant sources are associated with any sources.

Secure virtualization - CSP can use an Advanced Cloud Protection system (ACPS) to ensure the security of guest virtual machines and of distributed computing middleware. Behavior of cloud components can also be monitored by logging and periodic checking of executable system files. Trust model for interoperability and security. There should be separate domains for providers and users, each with a special trust agent. An independent entity that gathers security information used to gather security information is a confidence agent. Check for an end point. For service providers and clients, there should also be various trust strategies (Worlanyo, (2015).

III. How an organisation should protect their data

1. Using reputable and quality Cloud Server service

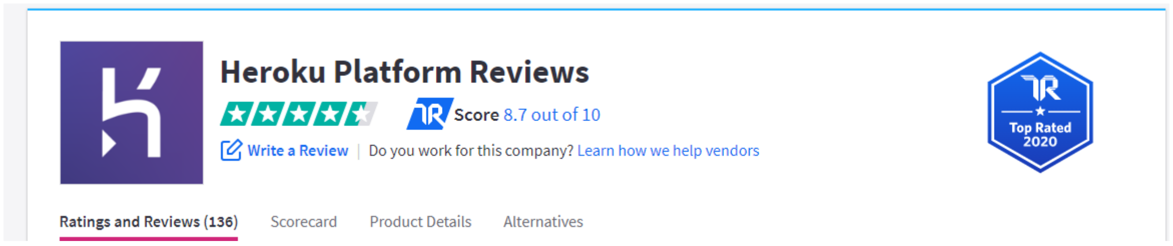
Cloud technology is developing day by day, security is also more complete. To ensure the safety of Cloud Server, the system has advanced security in many layers, ensuring safety for data in and out of the system, ... However, hacker attacks are also more and more sophisticated. ITs and businesses need to update the most modern security technology and software to ensure data safety on the Cloud. Use reputable and quality Cloud Server services so that vulnerabilities are discovered and patched in a timely manner, ensuring data safety.

For example: when a new updated version, there are so many hackers will try to break its security system. Therefore, the software will always need to have updated patches.

2. Choose a reputable provider

Choosing a reputable provider is considered a top priority when using Cloud Server. Because new reputable providers ensure your data is secure and secure. The IT team of the enterprise also needs to keep up with and fully update security methods to prevent network threats aimed at Cloud Server.

In this project, I have choose Heroku as service provider. According to the survey of (trustradius.com, 2020) Heroku get the score is 8.7 / 10 from 136 reviews. It's the score is quite high for a cloud service distributor.


Heroku Platform Reviews

Score 8.7 out of 10

Write a Review | Do you work for this company? Learn how we help vendors

Top Rated 2020

Ratings and Reviews (136) Scorecard Product Details Alternatives

Figure 35: Heroku Platform Reviews (trustradius.com, 2020)

In addition, there are many large and small businesses choosing heroku such as Toyota, Citrix, Westfield, Yesware, ...

3. Data encrypt

Data encryption can be considered as the last layer of defense of the business against outside threats. It is like a thief can break into your home but cannot unlock the safe. Data encryption in the cloud is the process of transforming or encoding data before moved to cloud storage (Olufohunsi, 2019). Typically cloud service providers offer encryption services - ranging from an encrypted connection to limited encryption of sensitive data - and provide encryption keys to decrypt the data as needed. Currently, vendors often support encryption services as a service, such as Microsoft Azure, Amazon Web Services (AWS), Salesforce, ... However, this may lead to a the number of incidents is not worth it. When the "encryption key" of the CSP can be stored together with the data. Therefore, businesses should adopt a BYOK - Bring your own key approach to ensure optimal data protection.

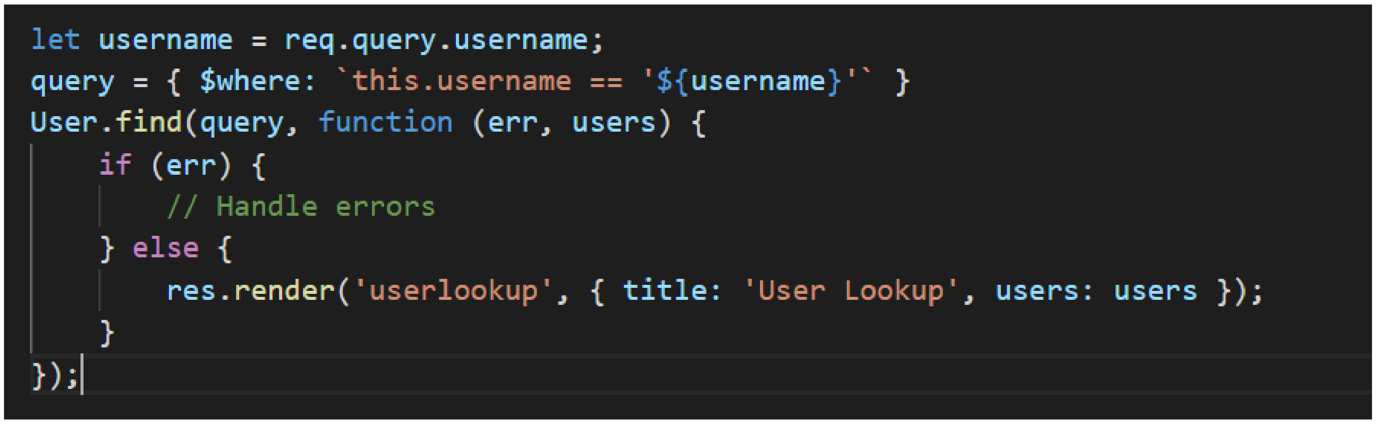
Take the example of Office 365, Microsoft's on-demand cloud application widely used by organizations across the globe to support employee mobility by providing anytime, anywhere access to applications. Microsoft email - MS Outlook and business utility applications like MS Word, Excel, PowerPoint etc

Gemalto's BYOK solutions (SafeNet ProtectApp and SafeNet KeySecure) for Office 365 not only ensure that organizations have complete control of their encrypted cloud data, but also support efficient management of encryption keys of other cloud applications such as Azure, AWS, Google Cloud, and Salesforce (Gemalto, 2020).

4. Protect data from client side.

After applying various protection measures to data. The dangers can still come from everyday tasks coming from the client side. We have two scenario here: When user access to the cloud with corporate computer. When user access to the cloud with personal computer. Security vulnerabilities can go to software in computers such as web browsers. In the first scenario, we can apply some security measures such as using anti-virus software or open source operating systems like linux. But for the second scenario, we need to apply some other different security measures like preventing sql injection, xss attack,…

For example: I will provide one of the instances that led to sql injection attack when doing the lookup. The (vulnerable) lookup code looks like this:



```
let username = req.query.username;
query = { $where: `this.username == '${username}'` }
User.find(query, function (err, users) {
    if (err) {
        // Handle errors
    } else {
        res.render('userlookup', { title: 'User Lookup', users: users });
    }
});
```

Figure 36: JS Injection Query String

This code may result in the object that, when executing the query, can retrieve the entire list of users. so they would like to pass in something which will always evaluate to true. If we pass in a string like '|| 'a' == 'a the query will become $ where: `this.username ==' '|| 'a' == 'a'` which of course always evaluates to true and thus returns all results. With JS injection, the database of the enterprise can be affected in many different ways.

Besides, the data entry also needs to be validated strictly according to the business rules. Protecting business data will include ensuring integrity and integrity of data.

5. Backup data

Most businesses today are not aware of the importance of backing up data as well as having backup plans when unfortunately something goes wrong with their website system. Operating system errors, software errors, malicious servers, corrupted hardware devices, hacker attacks ... are potential risks that can lose or damage important corporate data (information customers, email, books, transaction documents, contracts ...). Therefore, regularly backing up business data online is an urgent and extremely necessary job. However, storing backups on the cloud servers doubles the rate of attacks. Therefore the security measures I mentioned above should also be applied where the backup data is stored (Rouse, 2020).

D. SUMMARY

After this project, I have learned a lot more about cloud computing, including fundamental concepts about cloud computing like high performance computer, networking, and cloud architecture. Beside i have a chance to work with new technology are express-nodejs and mongodb. And my final product has applied cloud computing and deployed on heroku platform as PAAS model. In addition, problems that arise during cloud development such as security, .. have been applied in the article.

My website: https://asm2-gch18888.herokuapp.com/

My Source Code: https://github.com/LuxuryFi/Cloud/tree/master/MiniProject

References

Aslan, T., 2012. *Cloud physical security considerations.* [Online]
Available at: https://www.ibm.com/blogs/cloud-computing/2012/02/22/cloud-physical-security-considerations/
[Accessed 28 12 2020].

Chandra, C., 2017. *Data Loss Prevention.* [Online]
Available at: https://blogs.informatica.com/2017/05/03/data-loss-vs-data-leakage-prevention-whats-difference/
[Accessed 28 12 2020].

Gemalto, 2020. *Securing Access to Microsoft Office 365 with SafeNet Strong authentication from Gemalto.* [Online]
Available at: https://www6.gemalto.com/APAC-EN-201703-AuthOffice365-LP
[Accessed 28 12 2020].

Hein, D., 2019. *7 Cloud Storage Security Risks You Need to Know About.* [Online]
Available at: https://solutionsreview.com/cloud-platforms/7-cloud-storage-security-risks-you-need-to-know-about/
[Accessed 28 12 2020].

Nayyar, A., 2019. *Handbook of Cloud Computing.* 1st ed. s.l.:BPB Publication.

Olufohunsi, T., 2019. *DATA ENCRYPTION.* s.l.:s.n.

Rouse, M., 2020. *What is cloud backup and how does it work?.* [Online]
Available at: https://searchdatabackup.techtarget.com/definition/cloud-backup
[Accessed 28 12 2020].

Team, U., 2020. *What Are Cloud Leaks?.* [Online]
Available at: https://www.upguard.com/blog/what-are-cloud-leaks
[Accessed 28 12 2020].

trustradius.com, 2020. [Online]
Available at: https://www.trustradius.com/products/heroku-platform/reviews
[Accessed 26 12 2020].

Worlanyo, (2015. *A survey of cloud computing security: issues, chalenges and solutions..* s.l.:s.n.