

VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY  
THE INTERNATIONAL UNIVERSITY  
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING



# THESIS REPORT

## **AN E-LEARNING SYSTEM FOR SUPPORTING STUDENTS**

By

Nguyen Hoang Bao Khanh – ITITIU14045

A thesis submitted to the School of Computer Science and Engineering  
in partial fulfillment of the requirements for the degree of  
Bachelor of Information Technology/Computer Science/Computer Engineering

Ho Chi Minh City, Vietnam  
2018

# **AN E-LEARNING SYSTEM FOR SUPPORTING STUDENTS**

APPROVED BY: ADVISOR

\_\_\_\_\_,  
Nguyen Van Sinh, Ph.D, Chair

THESIS COMMITTEE

## ACKNOWLEDGMENTS

It is with deep gratitude and appreciation that I acknowledge the professional guidance of Dr. Nguyen Van Sinh. His motivation, enthusiasm, constant encouragement and support helped me to achieve my goal. During the preparation for thesis, my project as well as report has become more productive and encouraging due to Dr. Nguyen Van Sinh valuable instructions.

I would like to thank my family for their encouragement throughout my undergraduate study in International University and those who supported me to complete the research directly or indirectly.

Besides, my gratitude goes to my college friends and teachers who made these years a great learning experience. Especially friends in Computer Science department, they have helped me with researching new technology and showing me how to handle difficult in applying new technique to the thesis project as well as class project. Also, all teacher who were teaching me all the knowledge through 4 years in International University.

Last but not least, gratitude is also expressed to the members of reading and examination committee.

## TABLE OF CONTENTS

ACKNOWLEDGMENTS .....	3
TABLE OF CONTENTS .....	4
LIST OF FIGURES .....	6
LIST OF TABLES .....	7
ABSTRACT .....	8
CHAPTER 1 .....	9
INTRODUCTION .....	9
1.1. Background .....	9
1.2. Problem Statement .....	9
1.3. Scope and Objectives .....	10
1.4. Assumption and Solution .....	10
1.5. Structure of thesis .....	11
CHAPTER 2 .....	12
RELATED WORK .....	12
2.1. Technology Used .....	12
4.1.1. NodeJS .....	12
4.1.2. ExpressJS Framework .....	12
4.1.3. SocketIO Framework .....	12
4.1.4. MogoDB .....	13
2.2. Existing services using E-learning system .....	13
2.2.1 Online learning support. (Chegg) .....	13
2.2.2 Elearning: Advices about learning English .....	14
2.2.3 Elearning Center Page .....	17
CHAPTER 3 .....	20
METHODOLOGY .....	20
3.1. Overview .....	20
3.2. Use Case Diagram (chỉnh lại sau) .....	20
3.3. Activity Diagram .....	25
3.4. Database Structure .....	26
CHAPTER 4 .....	27
IMPLEMENT AND RESULTS .....	27
<b>4.2. Implement .....</b>	<b>27</b>
<b>4.2.1. Visual Studio Code .....</b>	<b>27</b>
<b>4.2.2. NodeJS .....</b>	<b>27</b>
<b>4.2.3. NodeJS Package .....</b>	<b>31</b>
<b>4.2.4. MongoDB .....</b>	<b>32</b>

<b>4.3. Results</b> .....	36
4.3.1. Create “room” with SocketIO. ....	36
4.3.2. Live Stream (1-many connection).....	37
4.3.3. Screen capture API.....	38
4.3.4. Firebase Connection.....	39
4.3.5. Real time chat application.....	40
CHAPTER 5 .....	41
CONCLUSION AND FUTURE WORK.....	41
5.1. Conclusion .....	41
5.2. Future Work.....	41
REFERENCES .....	42

## LIST OF FIGURES

*Figure 1 - Event-loop mechanism*  
*Figure 2 – Chegg*  
*Figure 3 – Discuss Forum*  
*Figure 4 - Online Course Register*  
*Figure 5 - Online Consultant registration*  
*Figure 6 – Online Quiz*  
*Figure 7 – Update news*  
*Figure 8 - Online Support*  
*Figure 9 - Online Course Register*  
*Figure 10 - Opening Schedule*  
*Figure 11 - Learning Schedule*  
*Figure 12 - Exam Information*  
*Figure 13 – System Architecture*  
*Figure 14 – Application Structure*  
*Figure 15 – Use Case Diagram*  
*Figure 16 – Activity Diagram*  
*Figure 17 – Entity-Relationship Diagram*  
*Figure 18 – Visual Studio Code*  
*Figure 19 – NodeJS*  
*Figure 20 – Introduction Installation of NodeJS*  
*Figure 21 – License Agreement*  
*Figure 22 – Destination Select*  
*Figure 23 – Installation Type*  
*Figure 24 – Summary*  
*Figure 25 – Checking the Installation*  
*Figure 26 – NodeJS Package*  
*Figure 27 – Homebrew*  
*Figure 28 – Installation of Homebrew*  
*Figure 29 – Installation of MongoDB*  
*Figure 30 – Run MongoDB*  
*Figure 31 – Testing MongoDB*  
*Figure 32 – mlab*  
*Figure 33 – mlab Provider*  
*Figure 34 – mlab Region*  
*Figure 35 – mlab Confirm*  
*Figure 36 – mlab Interface*  
*Figure 37 – mlab connect API*  
*Figure 38 – List Of Course*  
*Figure 39 – Example Creating room*  
*Figure 40 – Emit Event create room*  
*Figure 41 – Server listen to Event*  
*Figure 42 – Teacher Live Page*  
*Figure 43 – Connect User Camera*  
*Figure 44 – Server emit back Event Stream*  
*Figure 45 – Listen data from server*  
*Figure 46 – Twitch API*  
*Figure 47 – Firebase Config template*  
*Figure 48 – Get Download URL*  
*Figure 49 – Emit User-chat event*  
*Figure 50 – Emit Data to channel*

## LIST OF TABLES

*Table 1 – Use Case 1: Login/Logout*

*Table 2 – Use Case 2: Course Registration*

*Table 3 – Use Case 3: Add course*

*Table 4 – Use Case 4: Edit Cours*

*Table 5 – Use Case 5: Start Lecture*

*Table 6 – Use Case 6: View List of Students*

*Table 7 – Use Case 7: Register*

*Table 8 – Use Case 8: Join Lecture*

*Table 9 – Use Case 9: Manage Users*

## ABSTRACT

The education is growing by developing many new reforms in recent years. One of those development is gradually transforming the traditional way of learning into online learning. There are lots of website that applied online learning into their system and gradually replaced the traditional way of learning. Online learning is a new technique which is not only help student to explore more knowledge in their leisure time and dig deeper while finding new sciolsim, but also save time for them to do other things. By saying this I mean, student can stay at home and learn whenever there are Internet connection, they are not required to go to teaching center to study.

To manipulate the convenient of this kind of system, in this thesis, I would to research and create a web-application that can help students in learning. Therefore, I will make a live streaming for student to learn directly, not watching the recorded video on website. This system will be called “An e-learning system for supporting students” which will help students save time in learning new knowledge and can consult with teacher.



# CHAPTER 1

## INTRODUCTION

### 1.1. Background

To catch up with new advances technology in web development, I decided to use NodeJs to implement back-end with 2 supported frameworks (Express and SocketIO), EJS template to control “View” folder in NodeJS (EJS like a place to show web Interface to User) and NoSql database to increase the speed of website as well as real-time communication between client and server such as:

- MongoDB: to store all information on website. (In this project, database will be deployed on online storage which is mlab.com, this website is the leading Database-as-a-Service for MongoDB).
- Firebase: to store user file or image (Firebase is considered as backend-as-a-Service).

There are courses that help me to fulfill those requirements such as: Web Application Development, Object-Oriented Programming, Principles of Database Management, etc.

In **Object-Oriented Programming** course helps me with thinking in new way of creating inherited class for multiple purposes such as querying data from NoSql database to use in the project and how the inheritance or others attribute of OOP work in class-based object-oriented languages.

Furthermore, **Principles of Database Management** only teaching me how to develop database, but database is also an important thing to notice in web development. If the developer did not design a database that can be maintained in the future in case the user wants to evolve the business, then the developer must re-design the database and sometime may affect the project which will cause some risk.

To sum up all the knowledge from above course and other courses, I used it in course **Web Application Development** to develop a full functional website for ecommerce, selling product online or emulating the library for student to buy or borrow books.

Moreover, I want to develop a system for student to learn online via Internet and study live with real-time communication but not uploading the course on website and student will view when they have free time.

### 1.2. Problem Statement

The traditional way of learning is used until the new way which is online learning becoming new trend of learning in recent year. There are several advantages and disadvantages of online learning way compare to the traditional way.

First, the traditional way of learning is being used by every school very long time ago and it would cause some problems like: waste of time, student can not go to study center due to heavy rain or sickness. These problems will lead to missing some important lecture involved in the final exam or quick test. In contrast, students can have a chance to discuss with teacher after the lesson or during break time. Moreover, going to class can helps student to expand their relationships with other students as well as teacher.

Second, online learning can solve the problem of staying at home but still able to catch up with new lesson. But, the online learning platform nowadays is only focusing on showing users the recorded video without having communication with the learners or teacher can answer student’s question through comments in that lecture. By this I mean, the hold course video has

been developed and lesson video has already been recorded so there might be a chance that the knowledge/ technique is out of date.

Finally, this project is about creating an online learning for student to interact with teacher when the lesson is live on the Internet and teacher can teach the lesson as well as answer the question of student during the lecture.

### 1.3. Scope and Objectives

This project will consist of creating an application for any type of platform which used teacher and student model such as school, teaching center, etc. For any of the model above, they have the same type of user: staff, teacher, student. Each type above has their own function in the system:

- For staff(admin): Create course, teacher. Manage course, user information. Course will be created based on faculty list of new semester course.
- Teacher: view assigned courses, control class information(number of student,upload lecture slide,live lesson as well as online chat).
- Student: register account, course. Join lesson and get slide from registered course.

### 1.4. Assumption and Solution

There are lots of limitation to complete the full project of supporting student in studying online. Until now, I had encountered some problems such as: applying SocketIO framework to handle event from User and time limitation.

First, in the beginning of the project, I must research and test for applying live streaming in several language but the most suitable one is NodeJS because it supports more on client-server scripting compare to other languages. Furthermore, NodeJS applications mostly use **Single Threaded Event Loop Mode** architecture to handle multiple concurrence clients and it is an asynchronous operation. For example, the below code snippet shows how the Event-loop mechanism function:

Second problems which is also related to socket.io, whenever Teacher execute livestream function, the connection will stream to every course that belonged to that teacher. To fix this problem, Socket.io is a dream technique which can create unique room.

First two problems can be solved by using SocketIO Framework. For example, creating a event for User to listen like in Figure . Look Section to understand more about how SocketIO work

```
io.on("connection", function(socket) {
  console.log("Someone Connected " + socket.id);

  socket.on("creat-room", function(data) {
    socket.join(data);
    console.log(data);
  });
});
```

Figure 1 - Event-loop mechanism

Third is the time, the project is still in developing progress and use storage that can not hold the large amount of user connect at once or there might be a chance of having error while running. More functions will be developed in the future and the complete project will finish as soon as possible.

### 1.5. Structure of thesis

The thesis structure will be divided into 5 chapters: Introduction, related work, methodology, implementation and result, discussion and evaluation, conclusion and future work.

In chapter 1, I will introduce some background knowledge to help me with this project, some advantages and disadvantages of the traditional learning method and online learning system. Those parts are mainly about previous knowledge after going in to coding the project. The remaining is about designing a model for this project and some difficulties might be encountered while doing.

In chapter 2, it is mainly about all techniques which are available in this project and the researching of the market before designing or going into work. How to apply these existing technologies into the project, which system is already published in the market before and analyze all the software, advantages, disadvantages of those systems.

In chapter 3, mainly about diagrams of the app such as Use Case Diagram, Activity Diagram and include the way to apply methods and techniques mentioned in chapter 2.

In chapter 4, how to implement those relevant technicals in this project and some pictures to show the beta version of how this app will be structured when completing all the work.

In chapter 5, the final chapter will mention about my experiences and some lessons learned after the project is completed. Most important part is the future work to improve the application more effectively.

## CHAPTER 2

### RELATED WORK

#### 2.1. Technology Used

In chapter 2, I would like to review all the technology that I used to build the E-learning platform and some existed service on the market.

New languages are using data in the form of JSON due to its readability. JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including NodeJS, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

In this project, I will use several frameworks which support NodeJS to build back-end server and database which is familiar in using JSON as well as query JSON document as NoSQL database program like NodeJS language, Express and SocketIO framework, MongoDB database (online database website: mlab and firebase).

##### 4.1.1. NodeJS

NodeJS is an open-source, cross-platform JavaScript run-time environment that executes JavaScript code server-side. It was first written in 2009 by Ryan Dahl after the introduction of the first server-side JavaScript environment, Netscape's LiveWire Pro Web. He was desired an easier way to handle file upload progress bar on Flickr after seeing the browser did not know how much of the file had been uploaded and had to query the Web server. In 2008, NodeJS combine with Google's V8 JavaScript engine, an event loop, and a low-level I/O API.

NodeJS package was introduced to NodeJS in 2010, npm, is the most advantages in this project due to its ecosystem which is considered as the largest ecosystem of open-source libraries in the world. NPM designed to simplify installation, updating, and uninstallation of libraries.

Node.js is an asynchronous event-driven, non-blocking I/O model that makes it lightweight and efficient. Every single event clicked by user will fire the connection to server-side, NodeJS will work during the connection is fired otherwise NodeJS will sleep.

##### 4.1.2. ExpressJS Framework

There are several frameworks that Node supports but I choose Express to be this project frameworks. Express is a web application framework for Node.js, released as free and open-source software under the MIT License which is designed for building API.

The original author was TJ Holowaychuk who described Express as Sinatra-inspired server. Express is the backend part of the MEAN stack, together with MongoDB database and AngularJS frontend framework.

##### 4.1.3. SocketIO Framework

The main purpose of this project is to carry out the live connection between Student and Teacher to get the best result for Student. To obtain this, SocketIO is the perfect Framework which was built to create real time NodeJs application. There are four main functions of SocketIO which relates to Real-time:

- Real-time analytics: Push data to clients that gets represented as real-time counters, charts or logs.
- Binary streaming: Starting in 1.0, it's possible to send any blob back and forth: image, audio, video.

- Instant messaging and chat: Socket.IO's "Hello world" app is a chat app in just a few lines of code.
- Document collaboration: Allow users to concurrently edit a document and see each other's changes.

#### 4.1.4. MogoDB

MongoDB is a NoSQL database which uses JSON like documents with schemas, MongoDB is also a free and open-source cross-platform document-oriented database program. Moreover, there are lots of features that MongoDB support such as: queries, indexing, replication, load balancing, server-side javascript execution.

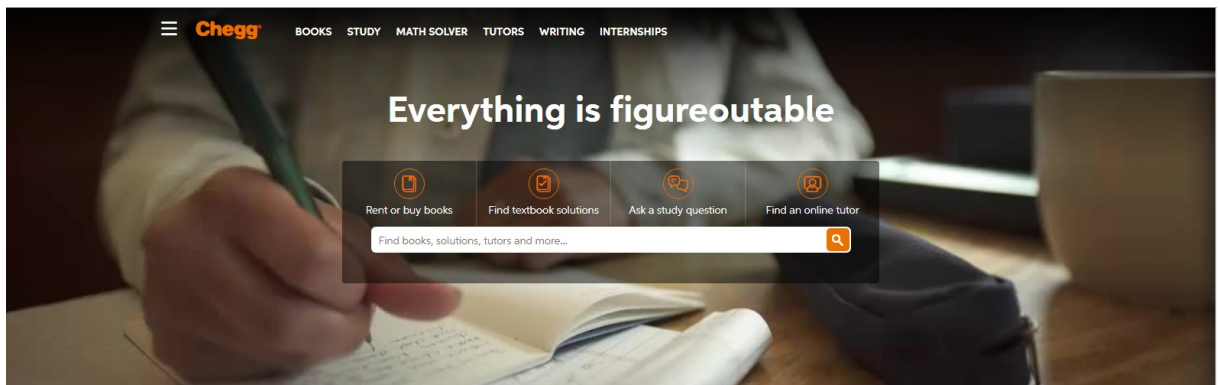
MongoDB can access through online storage such as: firebase, mlab, mongocloud.etc. These storage site will allow JSON data to be stored.

## 2.2. Existing services using E-learning system.

Online learning is not a hot trend anymore because there are lots of learning page existed on the market before. I will introduce three pages which has the same function for supporting student in learning online.

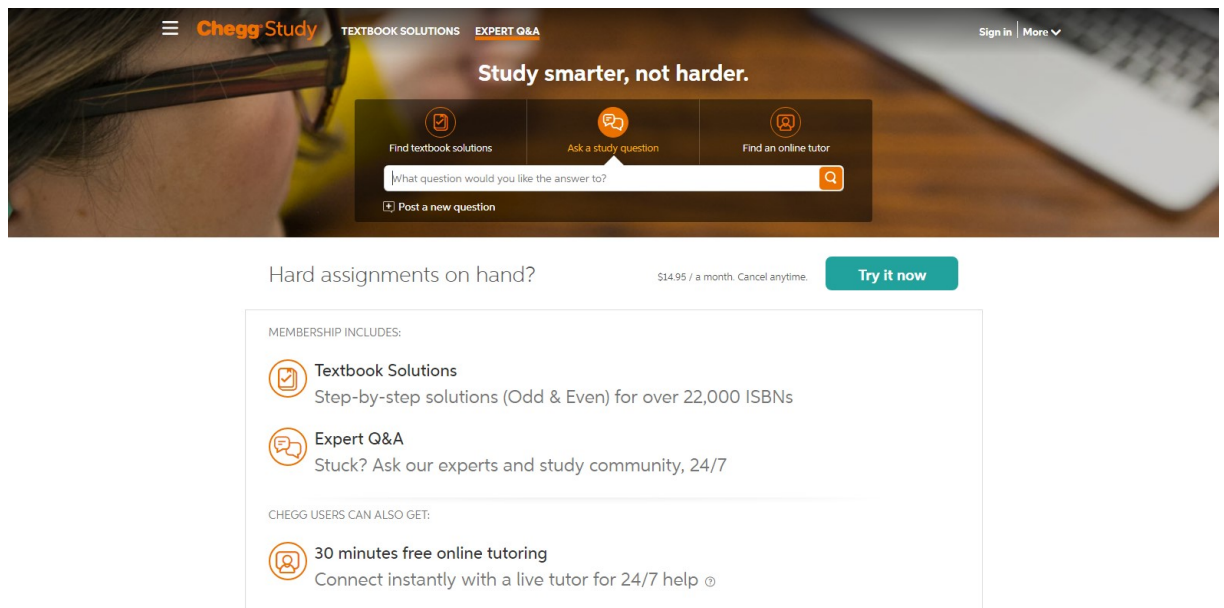
### 2.2.1 Online learning support. (Chegg)

- Website Reference: <https://www.chegg.com/>
- Functions:
  - Online tutoring: Student can choose: rent or buy books, find textbook solutions, ask a study question.



*Figure 2 – Chegg*

- Forum to discuss question with other student: Student can post their question and others who knows the answer can help.



*Figure 3 – Discuss Forum*

- Algorithms:
  - Client-Server.
- Advantages:
  - Student can learn with the best teacher in different college.
  - Get instant help when needed (no need to fill out forms, download any applications and software, or go through a series of phone calls)
  - More flexibility during the school week.
  - Many teachers in different fields to choose.
- Disadvantages:
  - Have to pay an amount of fee to chat with tutors.
  - Not include video chat directly to student only messenger.

### 2.2.2 Elearning: Advices about learning English.

- Website Reference: <http://mshoagiaotiep.com/>
- Functions:
  - Online Course Register: User can request a schedule for checking knowledge about course that they want to study.



*Figure 4 - Online Course Register*

- Online Consultant registration: Filled out the form so that consultant on mshoagiaotiep.com can contact back and answer question that user filled in.

*Figure 5 - Online Consultant registration*

- Tutorials
- Take online quiz: There are multiple quizzes/practises for student can choose and learn in advantage.

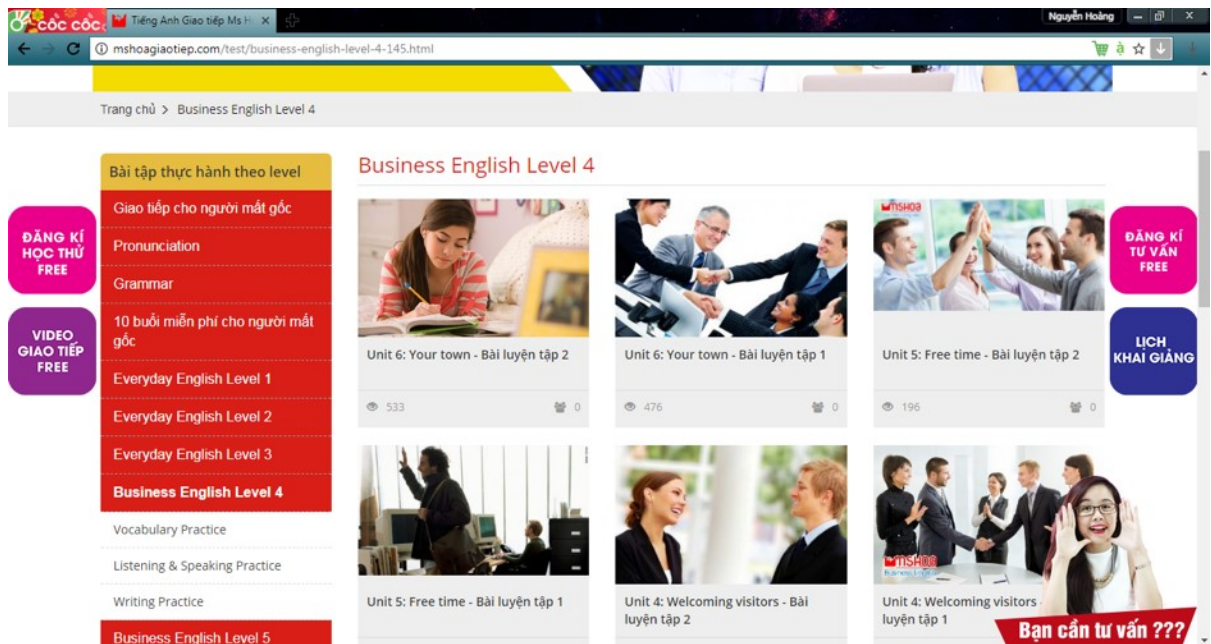


Figure 6 – Online Quiz

- Update news: News about class or English information will be post on main page of website.

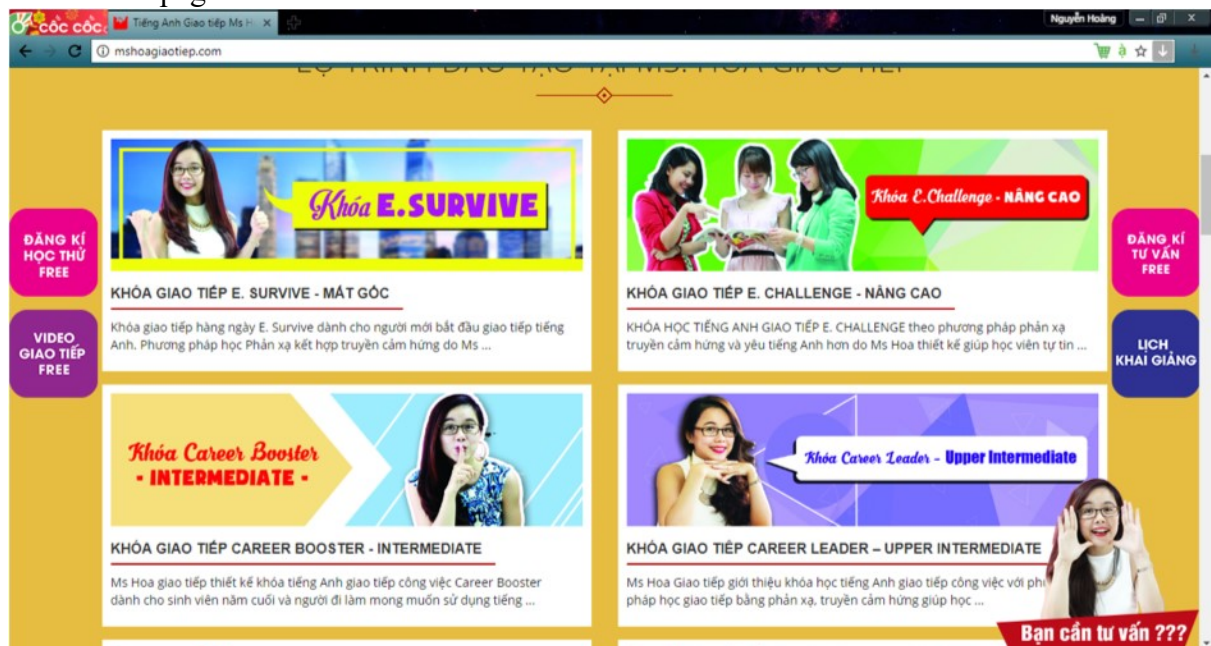


Figure 7 – Update news

- Algorithms:
  - Facebook messenger.
- Advantages:
  - Register online and then get help when consultant contact back via phone.
  - Take quiz and tutorial online to check knowledge.
- Disadvantages:
  - Not include video chat.



- Online support just for comment on facebook and then consultant will reply back (not direct chat).

### 2.2.3 Elearning Center Page.

- Website Reference: <http://elc.ehou.edu.vn/>
- Function:
  - Online Support via skype or phone: There are phone or button to connect skype for user to choose and make contact with consultant on <http://elc.ehou.edu.vn> website.

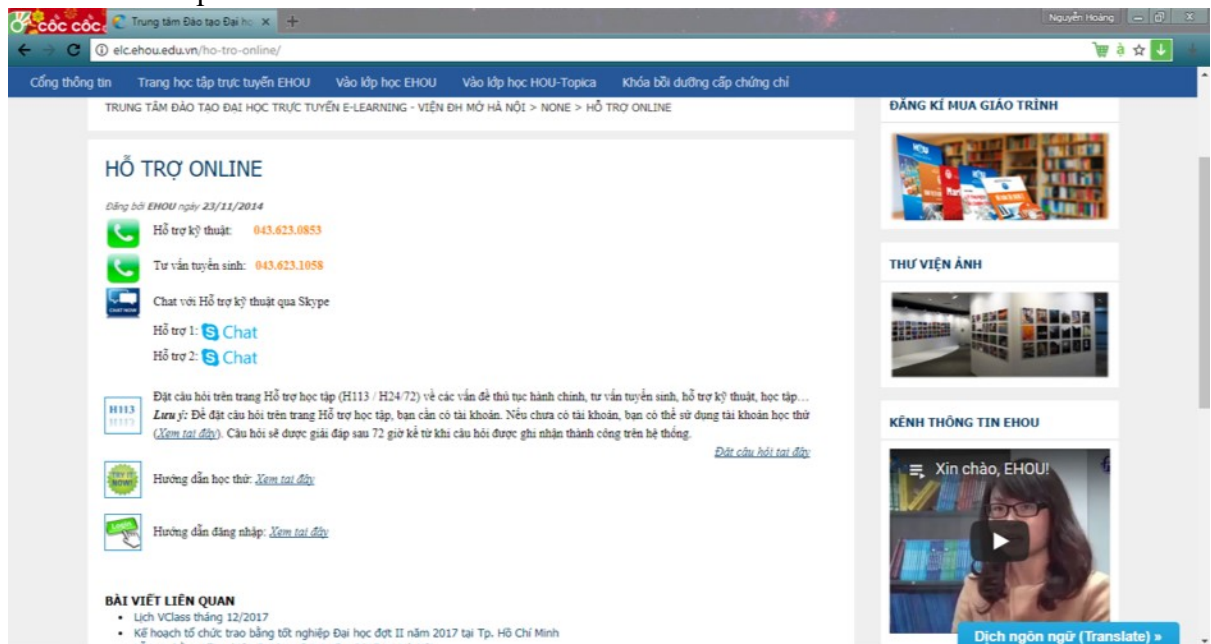


Figure 8 - Online Support

- Online Course Register: User can field their information about subject they wanted to study, etc so that consultant can base on that information and select course or contact back for more information.

**VIỆN ĐẠI HỌC MỞ HÀ NỘI THÔNG BÁO TUYỂN SINH ĐẠI HỌC TRỰC TUYẾN**

**CÁC NGÀNH ĐÀO TẠO**

- Kế toán
- Quản trị kinh doanh
- Tài chính ngân hàng
- Công nghệ thông tin
- Luật kinh tế
- Ngôn ngữ Anh
- Thông báo thi cấp chứng chỉ " ỨNG DỤNG CÔNG NGHỆ THÔNG TIN CƠ BẢN "(theo thông tư 03/2014/ĐTTT)

**ĐĂNG KÝ ĐỂ ĐƯỢC TƯ VẤN TRỰC TIẾP**

Họ và tên (\*):

Năm sinh:

Số điện thoại (\*):

E-mail (\*):

CMTND/Thẻ căn cước:

Văn bằng đã có(\*):

Địa điểm đăng ký học (\*):

Ngành đăng ký học/Chứng chỉ ứng dụng Công nghệ Thông tin cơ bản (\*):

**Đăng ký**

Bạn có thể liên lạc trực tiếp với chúng tôi qua số điện thoại: 024 3623 1058

Figure 9 - Online Course Register

- Schedule: Opening, Exam schedule.
  - Opening Schedule: Student who registered can check online for opening course.

**KẾ HOẠCH KHAI GIẢNG 2018**

**KẾ HOẠCH NHẬP HỌC, KHAI GIẢNG ĐÀO TẠO ĐẠI HỌC TỪ XA THEO PHƯƠNG THỨC E-LEARNING (EHOU) NĂM 2018**

Đợt	Khóa	Tháng	Ngày Khai giảng, nhập học (Dự kiến)
1	9	4	01/04/2018
2		5	27/05/2018
3		7	15/07/2018
1	10	9	09/09/2018
2		11	04/11/2018
3		12	23/12/2018

Ngoài lịch Nhập học và Khai giảng trên, sinh viên mong muốn theo học Kế hoạch của cá nhân vui lòng liên hệ với Cán bộ phận Tuyển sinh của Trung tâm Đào tạo Đại học E-Learning, Viện Đại học Mở Hà Nội để được hỗ trợ.  
Hotline: 0243 623 1058 Email: info@ehou.edu.vn

**THỦ TỤC NHẬP HỌC**

Căn cứ lịch Nhập học/ Khai giảng, Trung tâm E-learning thông báo cho sinh viên thời gian và địa điểm tổ chức Nhập học/ Khai giảng. Trong buổi Nhập học/ Khai giảng, sinh viên được phổ biến, hướng dẫn và làm các thủ tục nhập học:

- Công bố Quyết định công nhận sinh viên
- Phổ biến qui chế, qui định học tập
- Hướng dẫn về chương trình học tập và đăng ký kế hoạch học tập toàn khóa và kỳ 1
- Hướng dẫn qui định và thủ tục xét miễn môn
- Phát tài khoản học tập
- Thông báo lớp quản lý, Chủ nhiệm lớp Cố vấn học tập

Figure 10 - Opening Schedule

- Learning Schedule: Full information about learning schedule on each class or subject to make student can check easier.

**LỊCH HỌC EHO 2018**

**BẢNG DANH MỤC LỚP MÔN CÁC NGÀNH ĐÀO TẠO NĂM 2018 – CHƯƠNG TRÌNH EHO**

STT	Tên môn học	Mã môn	Số TC	Ngày bắt đầu	Ngày kết thúc	
1	Nhập môn internet và E-learning	EG38	4	01/04/2018	27/05/2018	<a href="#">Vào lớp</a>
2				27/05/2018	22/07/2018	
3				15/07/2018	09/09/2018	
4				09/09/2018	04/11/2018	
5	Phát triển kỹ năng cá nhân	EG35	4	04/11/2018	30/12/2018	<a href="#">Đăng ký</a>
6				23/12/2018	17/02/2019	
7				27/05/2018	22/07/2018	
8				15/07/2018	09/09/2018	
9	Tiếng Anh cơ bản	EN56	4	04/11/2018	30/12/2018	<a href="#">Vào lớp</a>
10				23/12/2018	17/02/2019	
11				15/07/2018	09/09/2018	
12				23/12/2018	17/02/2019	
13	Những nguyên lý cơ bản của CN Mác – Lênin	EG01	5	08/04/2018	27/05/2018	<a href="#">Vào lớp</a>
14				03/06/2018	22/07/2018	

Figure 11 - Learning Schedule

- Exam Information: School will post information about exam on each class.

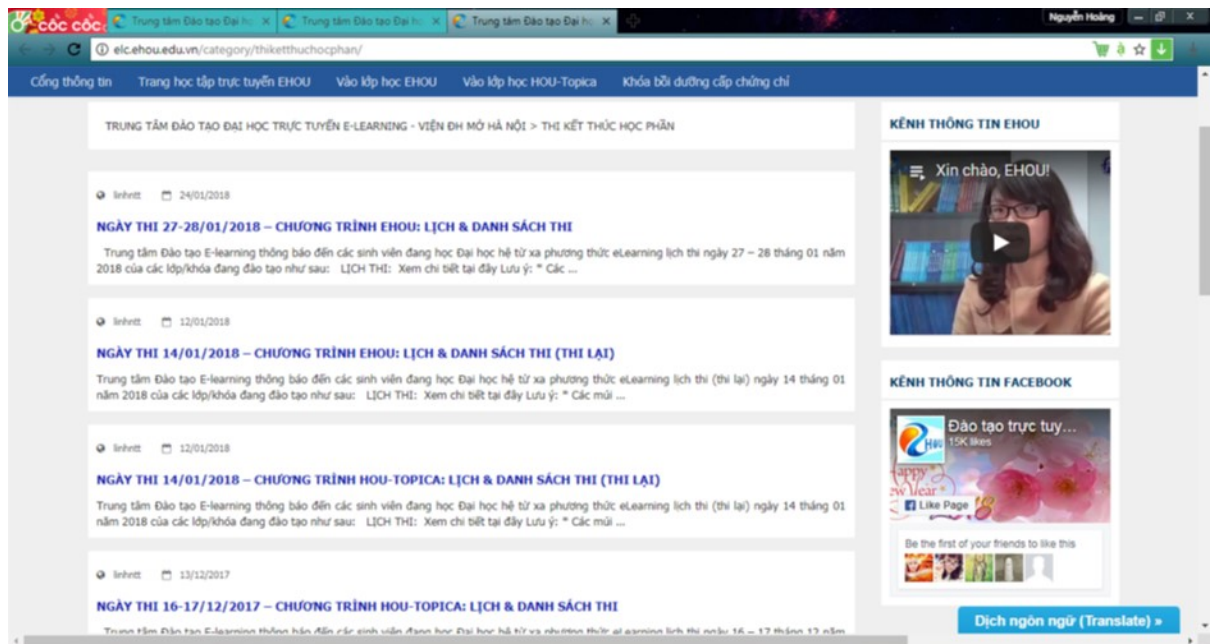


Figure 12 - Exam Information

- Tutorials.
- Algorithms:
- Advantages:
  - Student can register online and consultant will base on student's information to call for help later if they need.
  - Student can check schedule and news online instead of going to notice board on campus.
- Disadvantages:
  - Does not support online chat on website.

## CHAPTER 3

### METHODOLOGY

This chapter will introduce all the method that being used to build and some diagrams about Database, UI, etc.

#### 3.1. Overview

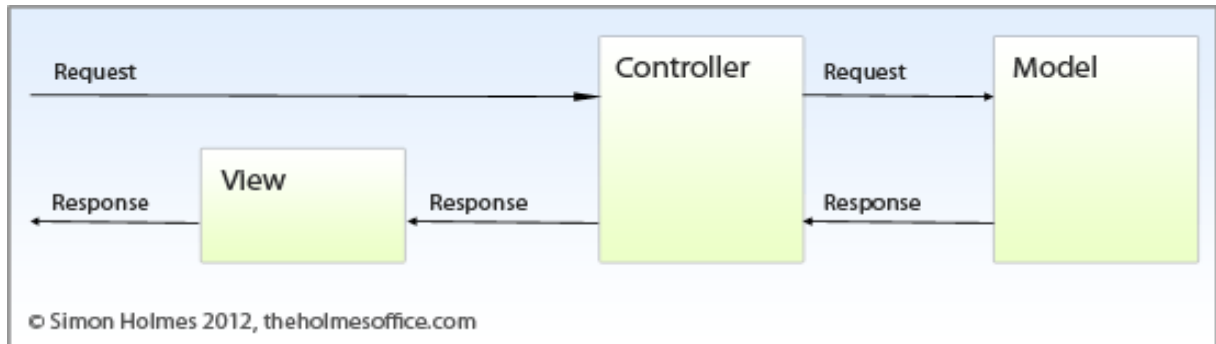


Figure 13 – System Architecture

The Figure 13 shows the process from the beginning (when user make a request) to the end (NodeJS return response). First, when user access to website url for example e-learning.com/login, the **Controller** (routes folder) redirect URL to corresponding routes to process users's request. After redirecting URL, Controller will make a request to **Model** (Model main function is to store, update/change, query data from MongoDB for Controller to manipulate, these files will be store in Models folder). Last step is returning query data to Controller and Controller will response that data to **View** (include ejs file to show data returned from Controller) to show the result to user.

The Firgure 14 will show the structure of this application based on MVC model.

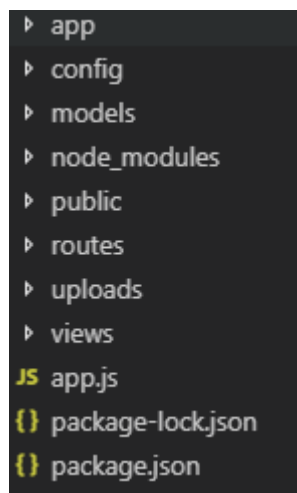


Figure 14 – Application Structure

#### 3.2. Use Case Diagram (chỉnh lại sau)

The figure 15 shows the basic function that actor (student, teacher, staff) can do in an e-learning system.

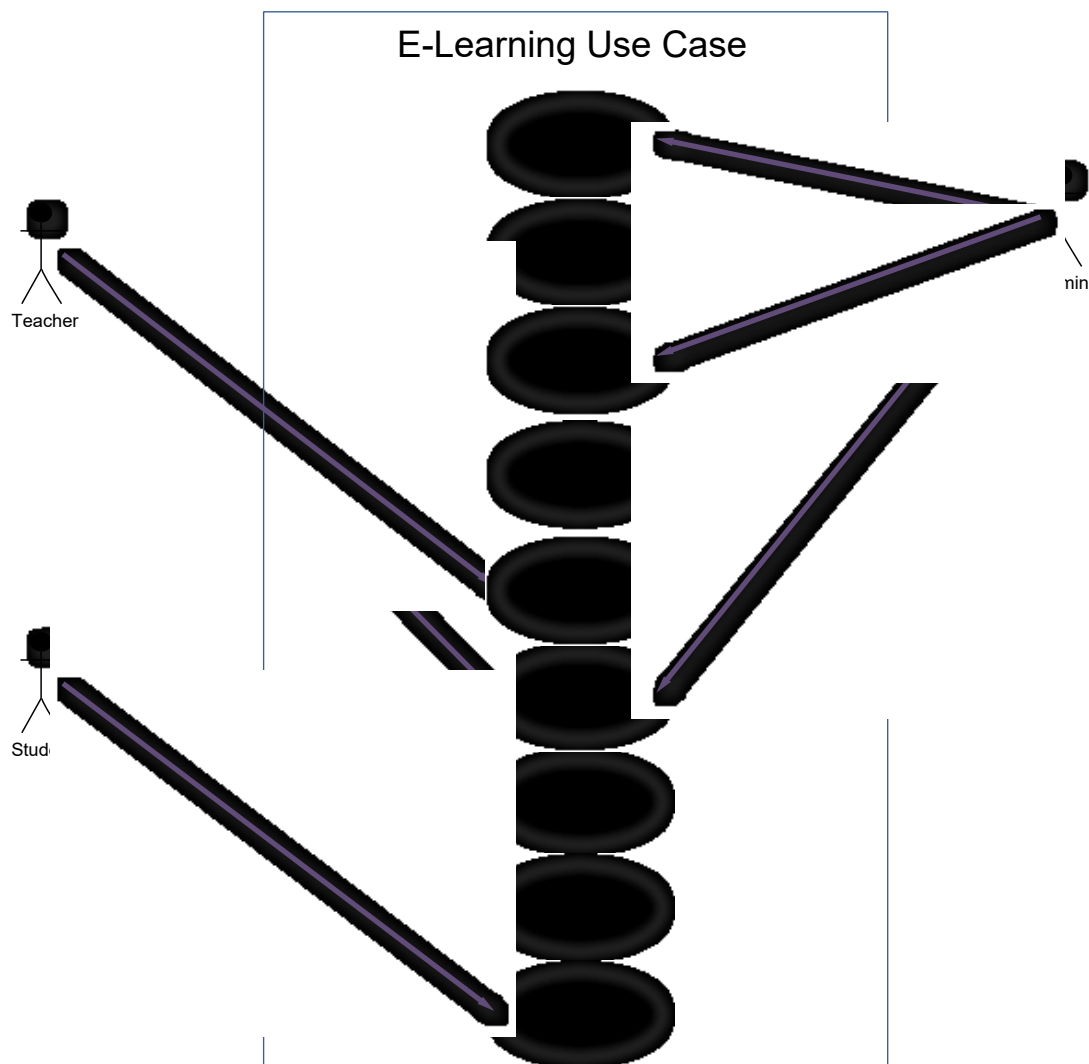


Figure 15 – Use Case Diagram

Description of use cases:

Table 1 – Use Case 1: Login/Logout

Name	UC-1: Login/Logout
Summary	Access permission to the system of users.
Rationale	All the users including staffs, lecturers and students need to login to use the functions of the system. Then they need to logout of their accounts to make sure that their information won't be leaked.
Users	Staffs, lecturers and students
Preconditions	Users must have their correct information (username and password) to access the system.
Basic Course of Events	<ol style="list-style-type: none"> <li>1. User goes to system site.</li> <li>2. User inputs username and password then click Enter.</li> <li>3. System will check the information and grant access permission if the information is valid. Otherwise, it will display error message to user interface.</li> </ol>

	4. User inputs information again if they have entered wrong information. 5. User logs in to the system and do the system's functions.
Alternative Paths	If the user doesn't do step 2 or step 4, they just can read the news of the university.
Postconditions	User gets access to the dashboard and can do the fuctions that system provides.

*Table 2 – Use Case 2: Course Registration*

<b>Name</b>	<b>UC-2: Course Registration</b>
Summary	Regist new course for new semester.
Rationale	Every new semester require student to regist course in order to learn. The course registration function allows student to regis new course and sever will save the course until the end of semester.
Users	Student
Preconditions	A list of course is loaded.
Basic Course of Events	1. User request for a list of course related to their field of study. 2. Software return a list related to user information. 3. User search for course that they want to learn in a limit of credit (from 12 to 24). 4. User click on "Save". 5. Software save user choice into database.
Alternative Paths	User may decide to abort the registration course during step 1,2 or 3. Software will return to Preconditions stage.
Postconditions	Software save the selected course by student in database.

*Table 3 – Use Case 3: Add course*

<b>Name</b>	<b>UC-3: Add Course</b>
Summary	Add more course to achieve full credit in one semester.
Rationale	Student may want to study more in one semester so that they can graduate on time.
Users	Staff.
Preconditions	List of course.
Basic Course of Events	1. User request for a list of course related to their field of study. 2. User find course they want to study and tick to the box next to course name. 3. User click "Save". 4. Software save user choice to database.
Alternative Paths	None
Postconditions	New course in the list of course.

*Table 4 – Use Case 4: Edit Course*

<b>Name</b>	<b>UC-4: Edit course</b>
Summary	Change the name of course, lecturer who teach this course, start time and finish time, cost for this course.
Rationale	This function support for staff when they have made some mistake while typing, so they can correct the mistake later.

Users	Staff
Preconditions	User login with staff account and course already add to database.
Basic Course of Events	<ol style="list-style-type: none"> <li>1. Staff click on edit button</li> <li>2. Then type the correct information for this course</li> <li>3 Click on “Done” button to finish this task</li> </ol>
Alternative Paths	In step 3, instead click “Done” button, staff can click “Cancel” button to cancel on process from step 1 to2. Database has not been changed.
Postconditions	Information have been changed in database.

*Table 5 - Use Case 5: Start Lecture*

<b>Name</b>	<b>UC-5: Start Lecture</b>
Summary	Teacher can start the lecture when he wanted
Rationale	Each course has starting time so that student can join in and learn, teacher can start the lecture.
Users	Teacher
Preconditions	User login with teacher account.
Basic Course of Events	<ol style="list-style-type: none"> <li>1. Click to the class which reached study time.</li> <li>2. Press “Start” to begin live stream.</li> </ol>
Alternative Paths	None
Postconditions	The camera begins to capture.

*Table 6 – Use Case 6: View List of Students*

<b>Name</b>	<b>UC-6: View list of Students.</b>
Summary	Teacher can know which class contained how many students in that class by viewing the list of students.
Rationale	There is different course that teacher can teach so list of students for teacher to know which student in which class.
Users	Teacher.
Preconditions	User login with teacher account. List of class has been loaded.
Basic Course of Events	Click on ‘View Class list’.
Alternative Paths	None.
Postconditions	None.

*Table 7 – Use Case 7: Register*

<b>Name</b>	<b>UC-7: Register.</b>
Summary	Student can register new account to learn.
Rationale	When user find some courses that they wanted to learn, register for an account is the first step after viewing those courses.
Users	User.
Preconditions	None.
Basic Course of Events	<ol style="list-style-type: none"> <li>1. Click on “Register” button.</li> <li>2. Input user information.</li> <li>3. Click “Done” button to finish this task.</li> </ol>

Alternative Paths	If some informations are not correct, then when user press “Done” it is not going to save to database, user must input again to correct it.
Postconditions	New user created.

*Table 8 – Use Case 8: Join Lecture*

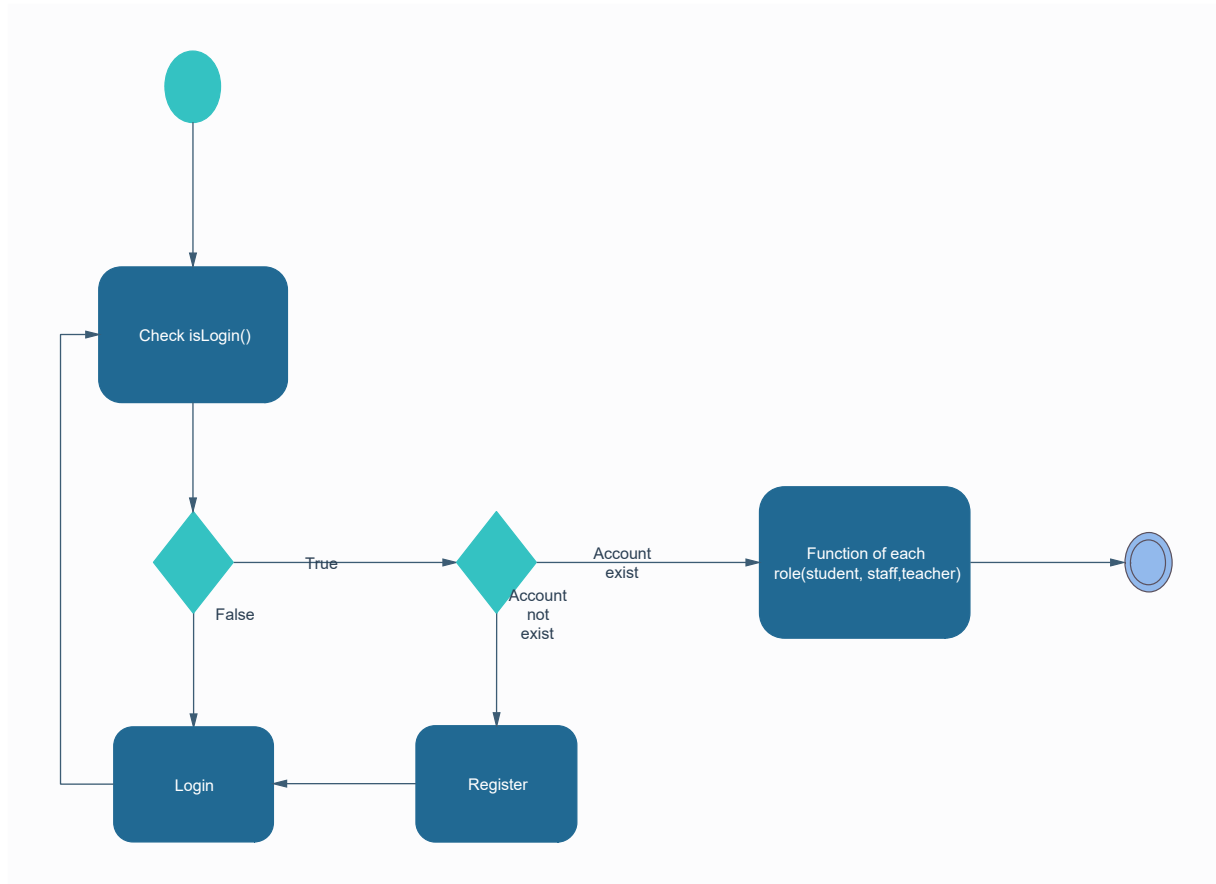
<b>Name</b>	<b>UC-8: Join Lecture</b>
Summary	When the lesson started, student can join whenever he wants to learn.
Rationale	When student login to this system, the list of registered class of Student will be shown and student will choose which class to view the live stream.
Users	Student.
Preconditions	User login with student account. List of class page loaded.
Basic Course of Events	Click on “Watch” button.
Alternative Paths	None.
Postconditions	Class page loaded.

*Table 9 – Use Case 9: Manage Users*

<b>Name</b>	<b>UC-9: Manage Users</b>
Summary	Create account for Student and Lecturer. Update necessary information for these user (Course, tuition fees)
Rationale	When the student login to our website to registration course or lecturer want to know information of course that they have to teach (room, time, student). In addition, staff ability to confirm about tuition fees.
Users	Staff
Preconditions	User login with staff account. Manage user page have been loaded.
Basic Course of Events	1. Click on ‘Create account’ 2. Input information for this account (username, password, role[student/lecturer/staff]) 3. Click “Done” button to finish this task
Alternative Paths	1. Instead click on “Create account”, we can click on “Update User” button to change user information. The do the same step 2 and 3 above. 2. In step 1, we can click on button “Remove” to remove this user account.
Postconditions	Database changed.



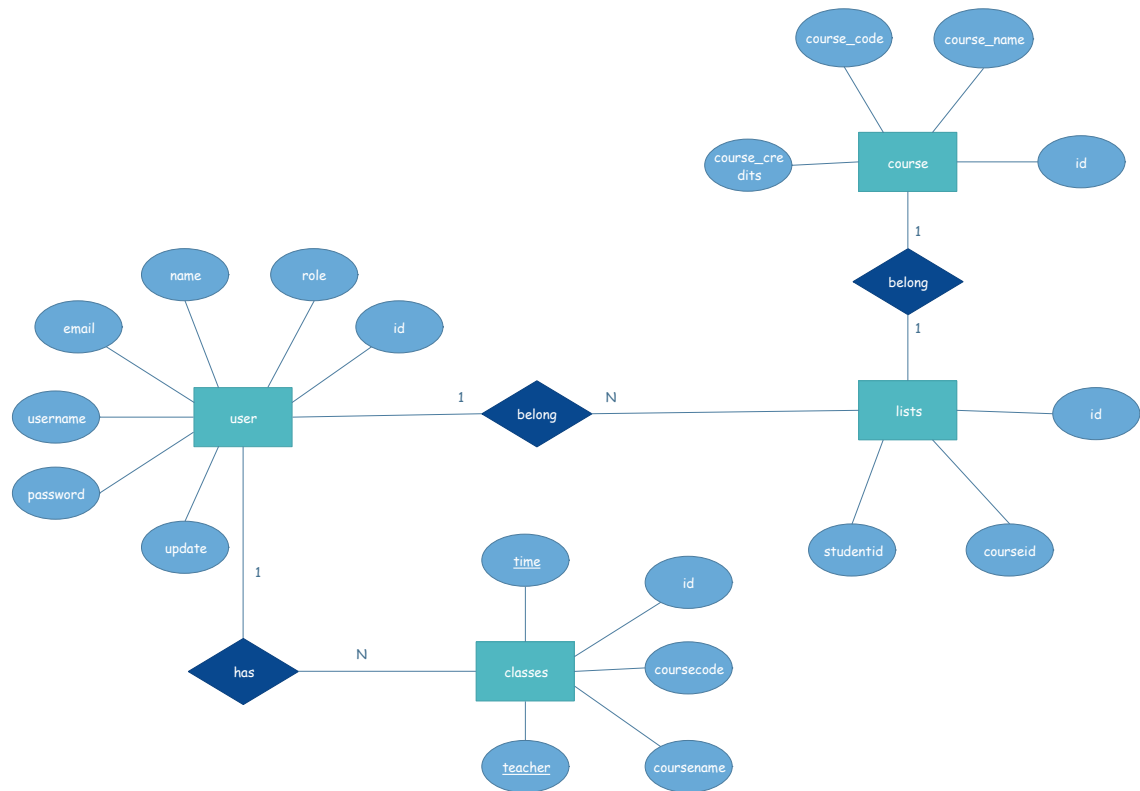
### 3.3. Activity Diagram



*Figure 16 – Activity Diagram*

The figure 15 is the activity diagram of my application project. When ever user Login into the system, it will check for the existence of that user in database. If login with correct information in database so user can continue to do with their own function (Eg: student can register course, watch stream. Teacher can live stream, view student list, Staff can create/add/edit new course to database or create class). And if the account is not existes so user can register for a new account and relogin.

### 3.4. Database Structure



*Figure 17 – Entity-Relationship Diagram*

The database of this application included four tables. First table is “**user**” which has the function of saving user information like name, id, email, etc. “**course**” is the seconde table, save all course information. Third table is “**lists**”, this is a middle table to save information of course and user with their id only, by using this each course contain different user list. The last table is “**classes**” which included all the information of a class like coursecode, coursename, etc. Each table have its own relationship, like “**user**” and “**classes**” is 1-to-many because 1 user can have many classes. Between “**user**” and “**lists**” is also 1-to-many because 1 user can be in many lists. The last relationship 1-to-1 which is different from others because 1 course can only be in 1 list (list of course is very limit not as many as classes due to 1 course can have many class).

## CHAPTER 4

### IMPLEMENT AND RESULTS

#### 4.2. Implement

##### 4.2.1. Visual Studio Code

Visual Studio is a code editor which I chose to implement this project due to its convenient ability. Visual Studio Code is redefined and optimized for building and debugging modern web and cloud applications. To use it, go to <https://code.visualstudio.com> and choose the version which suitable with your platform: MacOS, Windows or Linux.

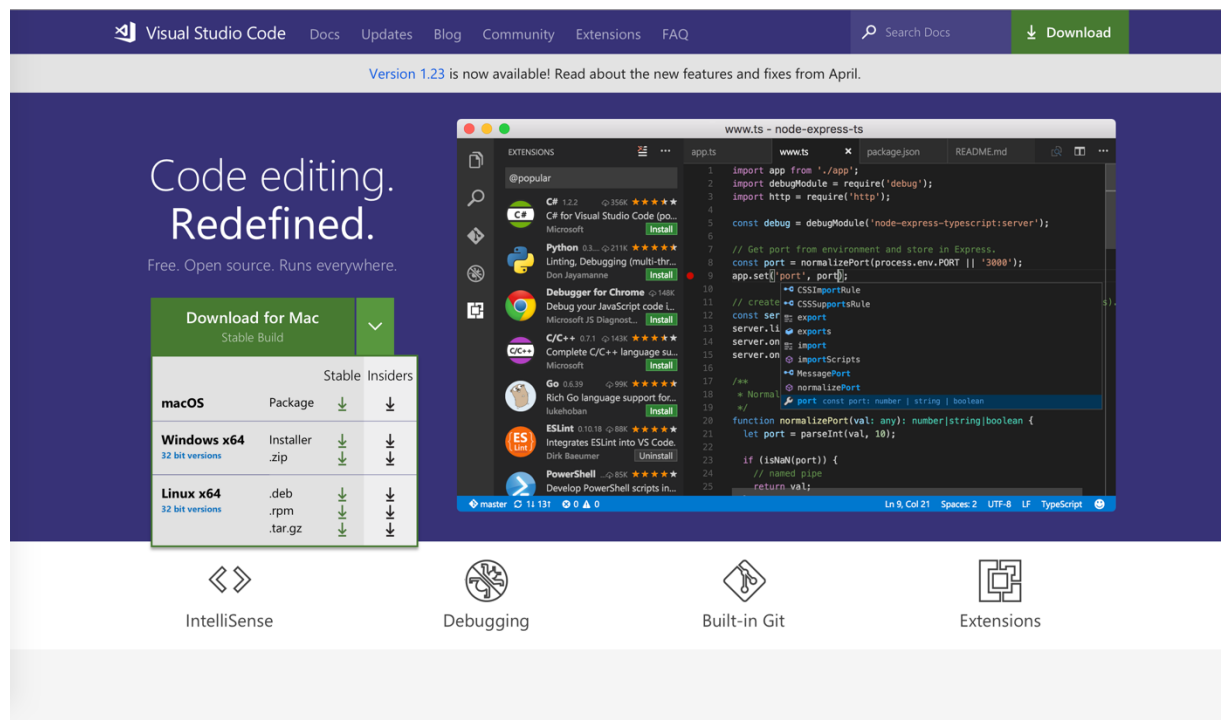


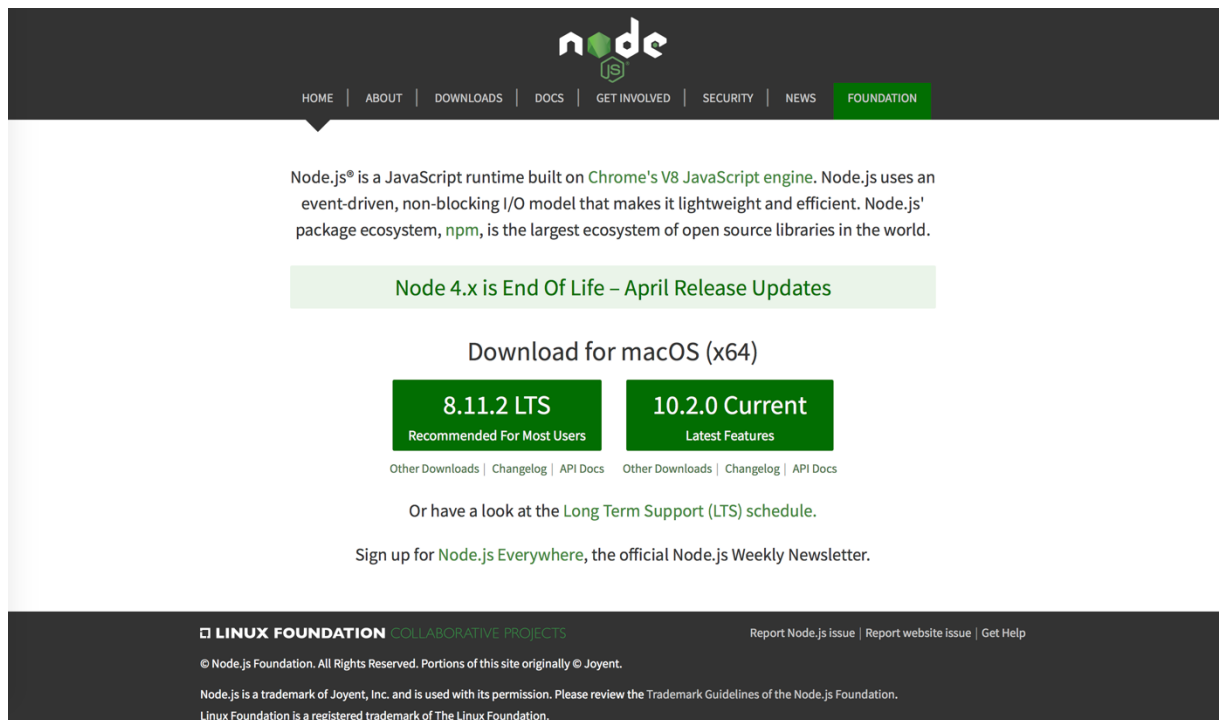
Figure 18 – Visual Studio Code

After downloading the software, it will install automatically into your machine.

##### 4.2.2. NodeJS

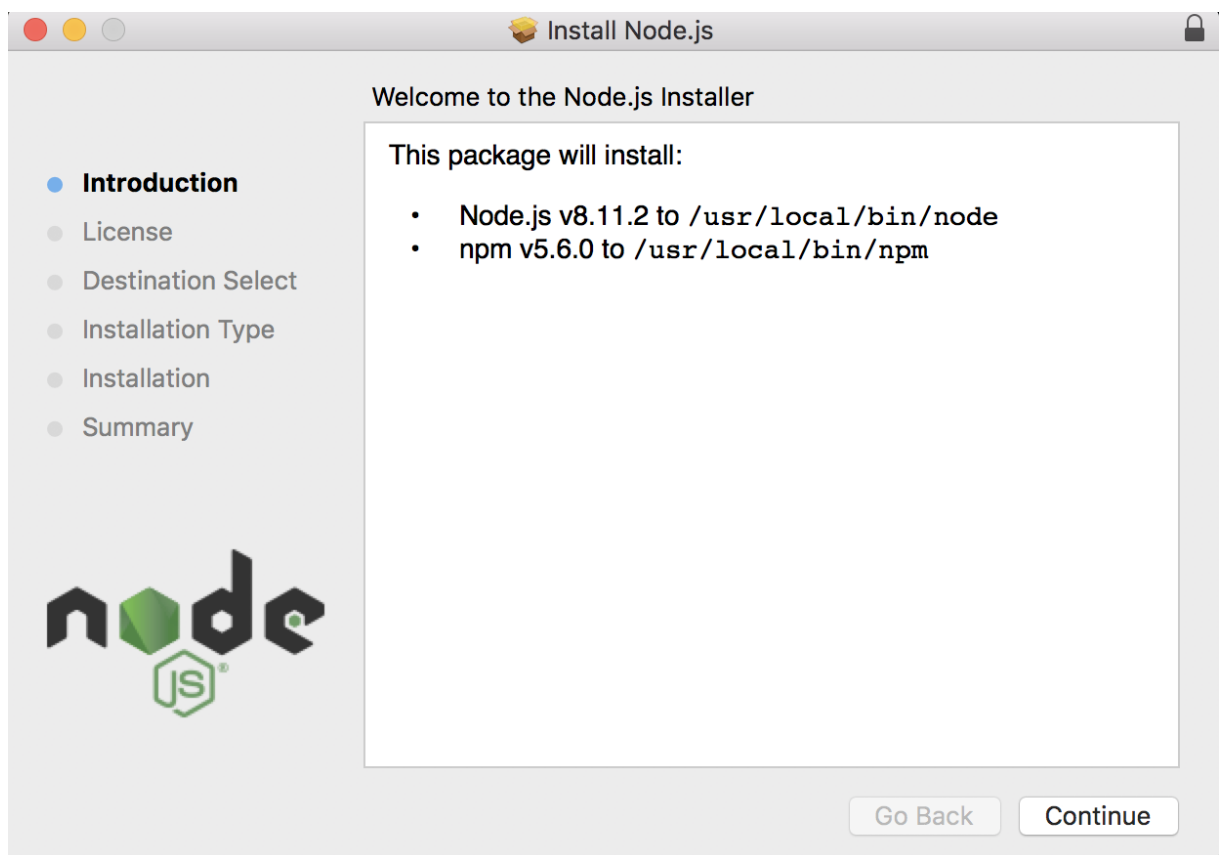
First, go to <https://nodejs.org/en/> and select the version for downloading. In my project, it is 8.11.2 LTS version because the newest version may not be stable for using.

After hitting download button, the software will be in your Download folder and should be ready to install when finishing downloading. Double Click on the downloaded software to install NodeJS to your computer.



*Figure 19 – NodeJS*

The installation of NodeJS is started. Press Continue to go to License part.



*Figure 19 – Introduction Installation of NodeJS*

Agree to the License to continue the installation of NodeJS.

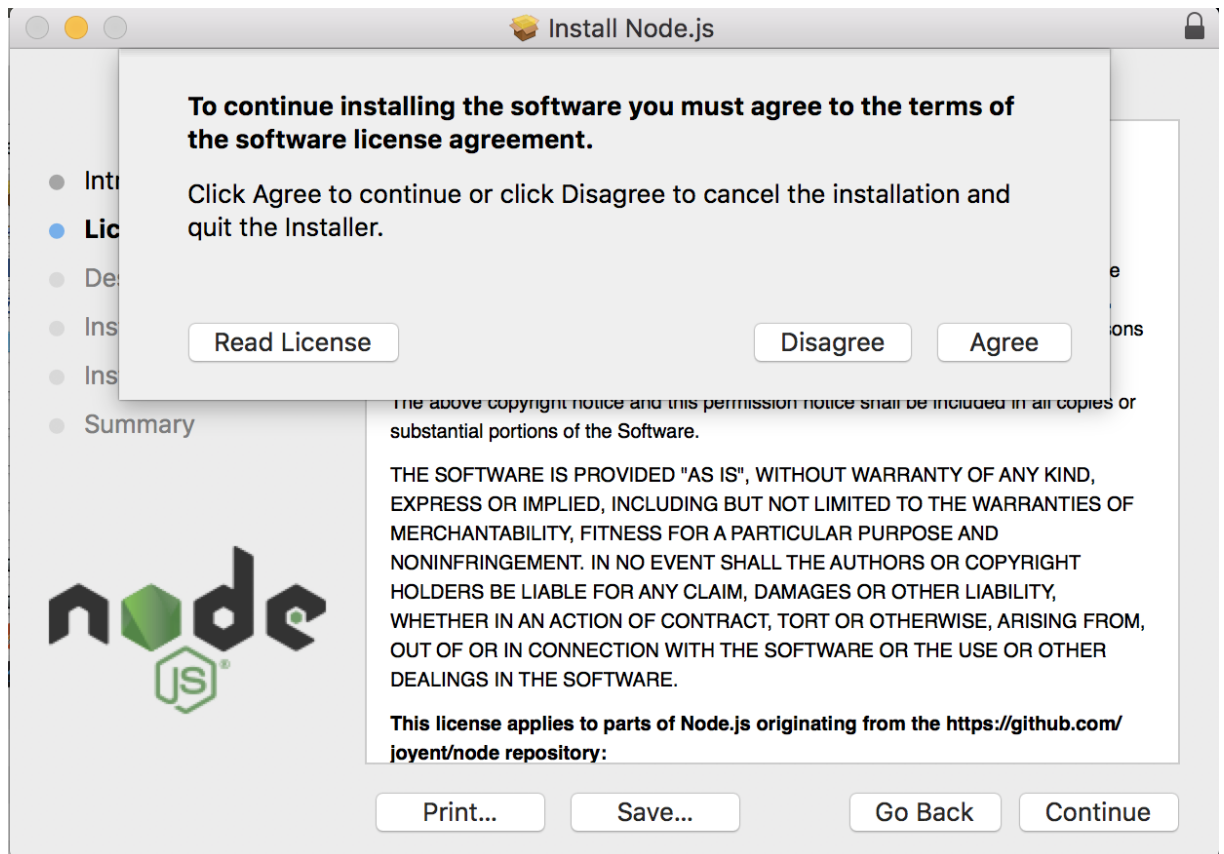


Figure 21 – License Agreement

Next step is choosing the folder to install.

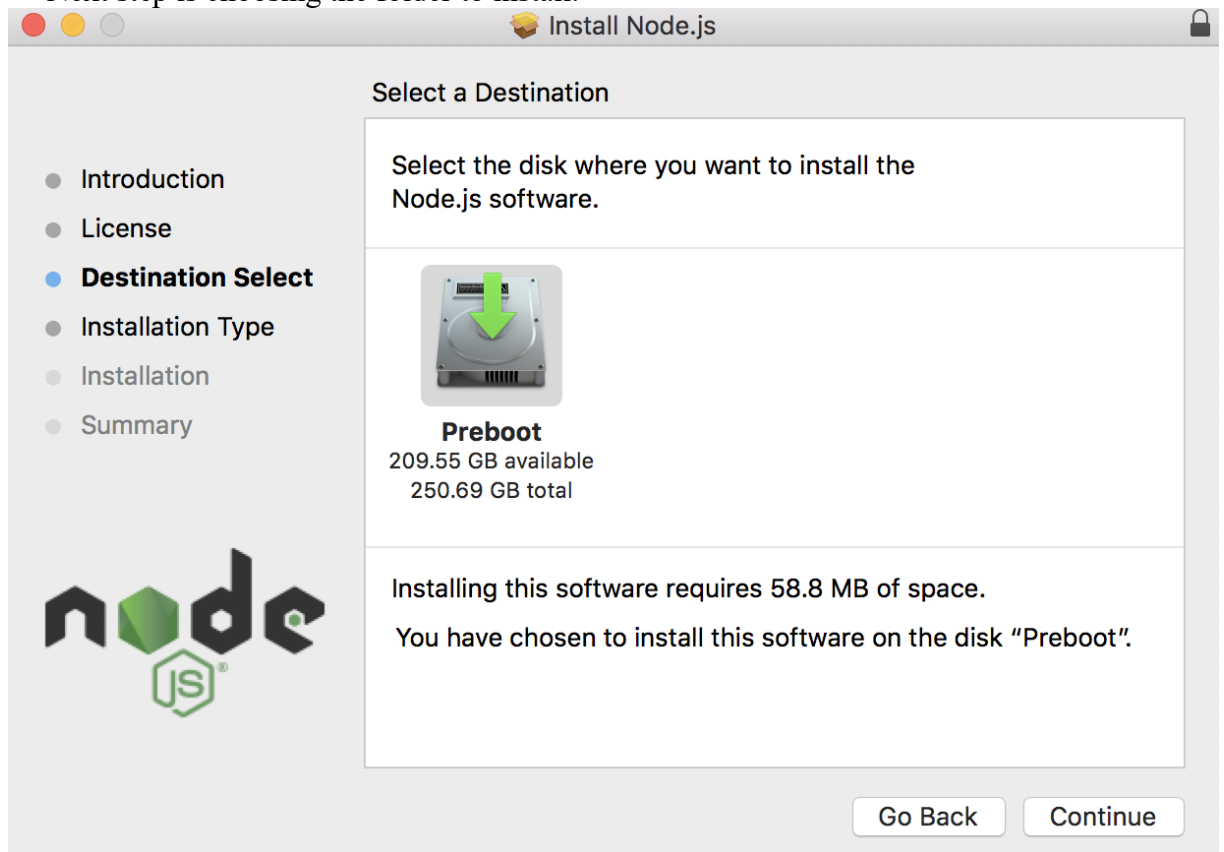
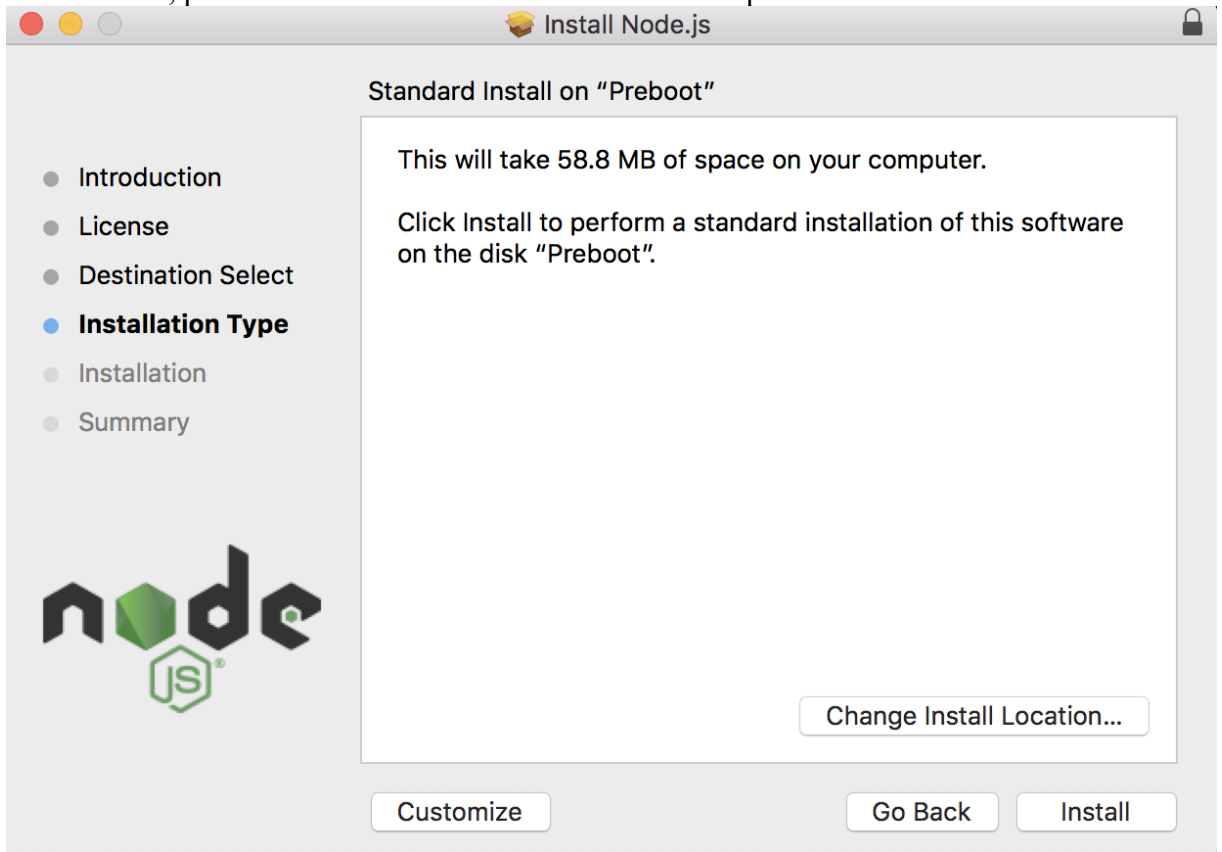


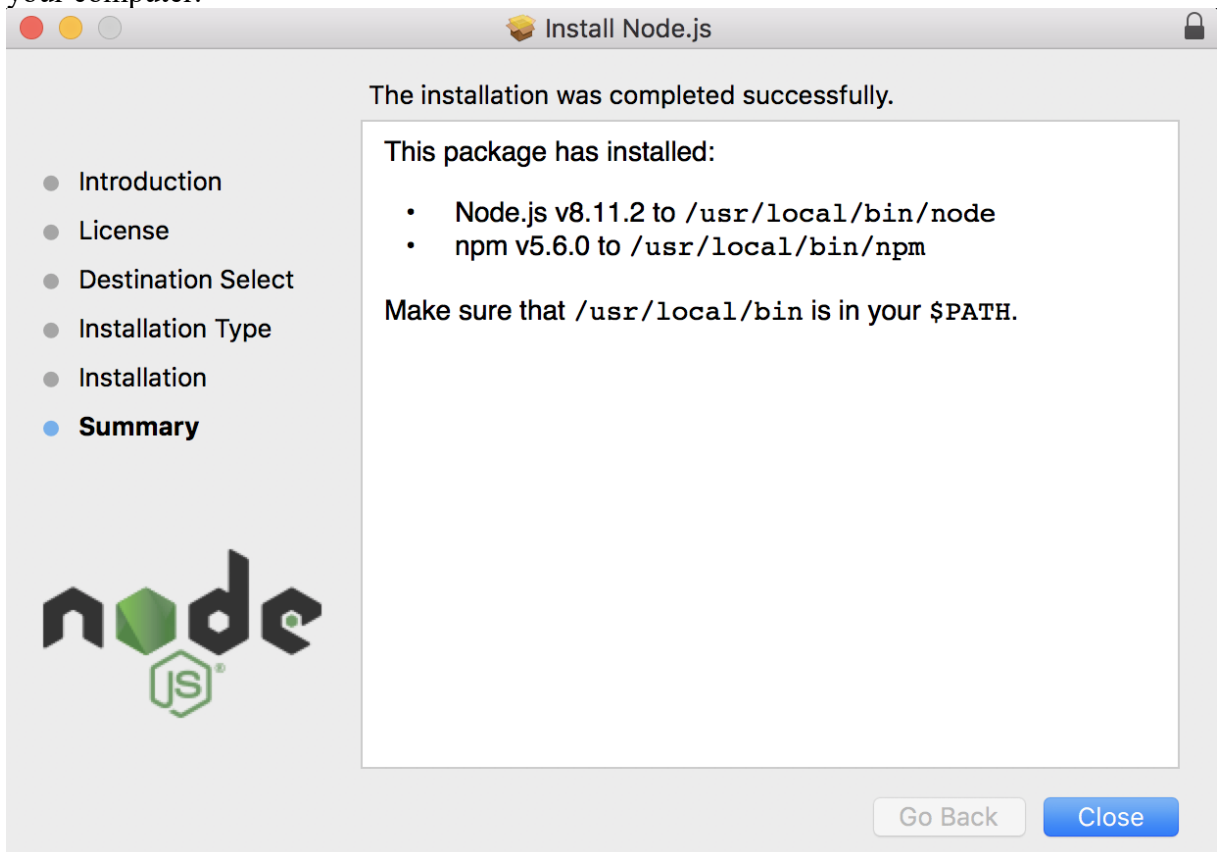
Figure 22 – Destination Select

After that, press continue and install to start the install process.



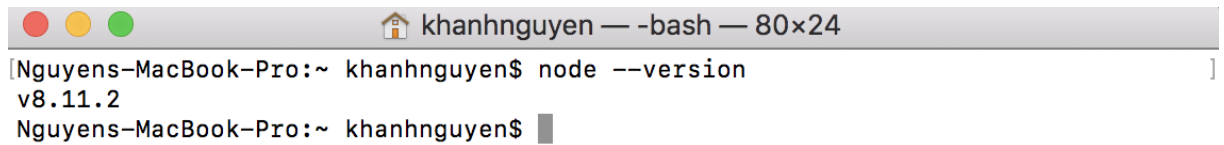
*Figure 23 – Installation Type*

The installation is quickly so the Summary part will show what you have just installed in your computer.



*Figure 24 – Summary*

Last step is to check whether you are installing NodeJS or not by opening the terminal and type `node --version`. If the terminal line shows like which means NodeJS has been installed successfully.

A screenshot of a macOS terminal window. The title bar shows three colored window control buttons (red, yellow, green) and the text 'khanhnguyen — -bash — 80x24'. The terminal content shows the command 'node --version' being executed, resulting in the output 'v8.11.2'. The prompt 'Nguyens-MacBook-Pro:~ khanhnguyen\$' is visible at the bottom.

```
Nguyens-MacBook-Pro:~ khanhnguyen$ node --version
v8.11.2
Nguyens-MacBook-Pro:~ khanhnguyen$
```

*Figure 25 – Checking the Installation*

### 4.2.3. NodeJS Package

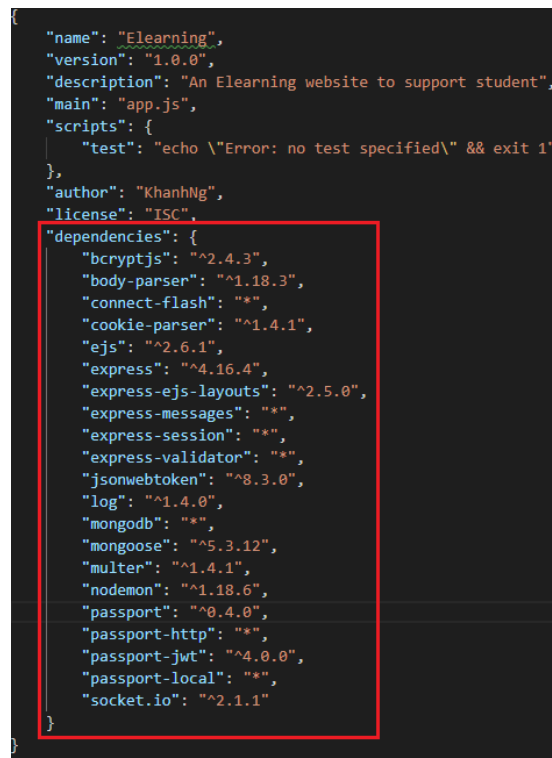
There are lots of package to support the project to run smoothly and those packages will help with the implementation. User can go to <https://www.npmjs.com> to explore more about package and how to use or install it. The basic command to install one package is “`npm install <package name>`”.

If the user wants to install as global so in others project may not have to install again then type “`npm install -g <package name>`”.

If the users only want to install the package for this project only so type “`npm install --save <package name>`”.

All the package after installing will be in package.json file and saved like the picture below. Dependencies category is all the package that programmer wants to use for the project.

Below Figure is all the dependencies which this project needed. All the dependencies key feature will be available on <https://www.npmjs.com/>, access to this website to find out more details about which function, key features and how to use the dependency.

A screenshot of a code editor showing the content of a package.json file. The file is for a project named 'Elearning' with version '1.0.0'. The 'dependencies' section is highlighted with a red rectangle and lists various packages like bcryptjs, body-parser, connect-flash, cookie-parser, ejs, express, express-ejs-layouts, express-messages, express-session, express-validator, jsonwebtoken, log, mongodb, mongoose, multer, nodemon, passport, passport-http, passport-jwt, passport-local, and socket.io.

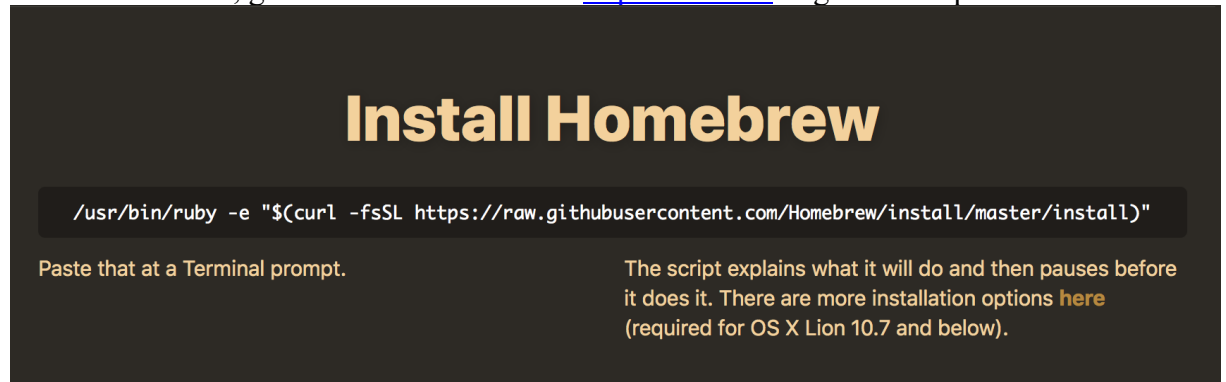
```
{
  "name": "Elearning",
  "version": "1.0.0",
  "description": "An Elearning website to support student",
  "main": "app.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "KhanhNg",
  "license": "ISC",
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "body-parser": "^1.18.3",
    "connect-flash": "*",
    "cookie-parser": "^1.4.1",
    "ejs": "^2.6.1",
    "express": "^4.16.4",
    "express-ejs-layouts": "^2.5.0",
    "express-messages": "*",
    "express-session": "*",
    "express-validator": "*",
    "jsonwebtoken": "^8.3.0",
    "log": "^1.4.0",
    "mongodb": "*",
    "mongoose": "^5.3.12",
    "multer": "^1.4.1",
    "nodemon": "^1.18.6",
    "passport": "^0.4.0",
    "passport-http": "*",
    "passport-jwt": "^4.0.0",
    "passport-local": "*",
    "socket.io": "^2.1.1"
  }
}
```

*Figure 26 – NodeJS Package*

#### 4.2.4. MongoDB

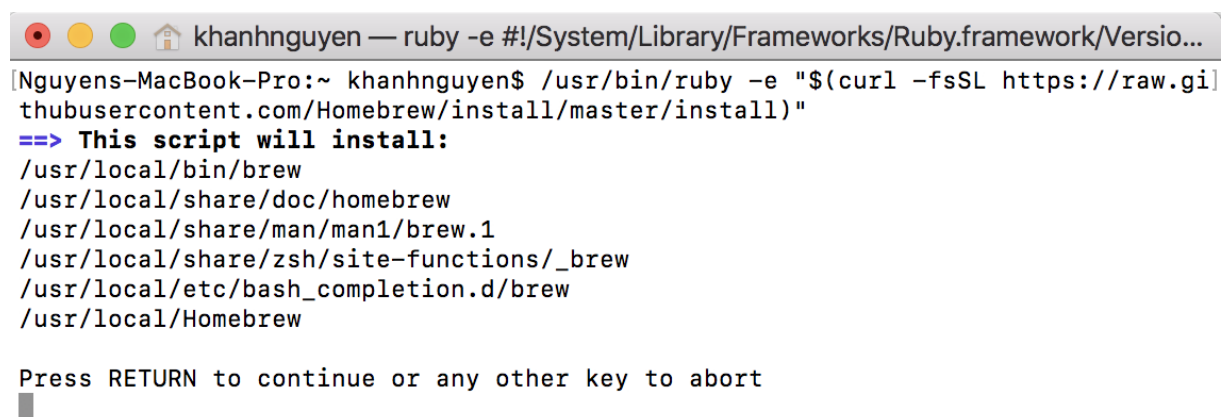
- Install Local Database

To install MongoDB on MacOS, the recommendation choice is to install through HomeBrew. First, go to HomeBrew website <https://brew.sh> to get the script:



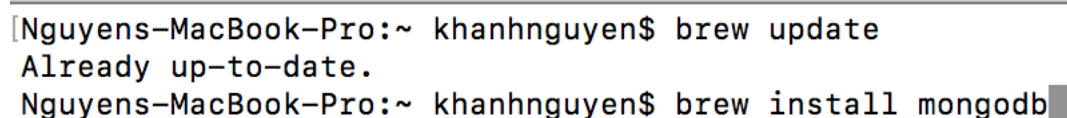
*Figure 27 – Homebrew*

Copy the script and paste to Terminal prompt and if press RETURN(Enter) button to continue the process. The password of your computer is required to continue the installation after pressing Return.



*Figure 28 – Installation of Homebrew*

The next step is following the instruction from MongoDB documentation page <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/>. Update the brew after the above process complete by using “brew update”. If it is uptodate then go to next step which is install MongoDB with “brew install mongodb”.



*Figure 29 – Installation of MongoDB*



The process is very quickly so it will take from 1 to 2 minutes. In order to run MongoDB, user must use root privilege to run it. By typing “sudo mkdir -p /data/db” , this step will create the directory to which the mongod process will write data.

To test whether the installation of MongoDB is finished or not, go to Terminal and type “sudo mongod” to run the server of MongoDB.

```
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten]
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten]
2018-05-26T09:52:09.045+0700 I CONTROL [initandlisten]
2018-05-26T09:52:09.045+0700 I CONTROL [initandlisten] ** WARNING: soft rlimits too low. Number of files is 256, should be at least 1000
2018-05-26T09:52:09.081+0700 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory '/data/db/diagnostic.data'
2018-05-26T09:52:09.082+0700 I NETWORK [initandlisten] waiting for connections on port 27017
```

Figure 30 – Run Mongod

Now it is running on port 27017 which mean the installation is successful. Next step, go to new terminal and type “mongo” which will start MongoDB database. And now user can have some command to check such as “show dbs” like the picture below to show all the database that existed in local machine.

```
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning.
2018-05-26T09:52:09.044+0700 I CONTROL [initandlisten]
2018-05-26T09:52:09.045+0700 I CONTROL [initandlisten]
2018-05-26T09:52:09.045+0700 I CONTROL [initandlisten] ** WARNING: soft rlimits too low. Number of files is 256, should be at least 1000
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
login      0.000GB
loginapp   0.000GB
testDB     0.000GB
>
```

Figure 31 – Testing Mongod

- Online Database:

To make this application work, programmer must use online database to store data and mlab is one of the databases which suits the best.

First, access to <https://mlab.com/> website to create new account.

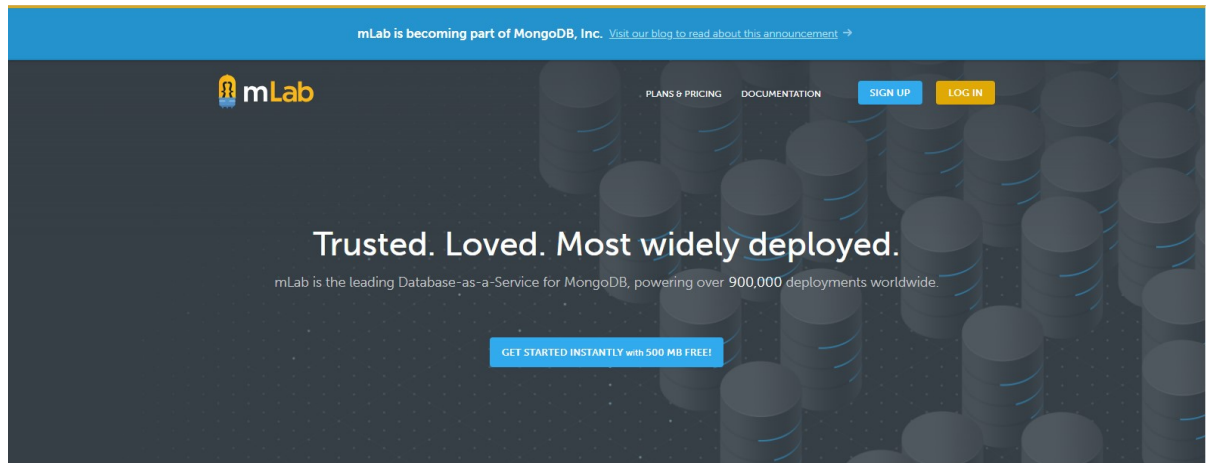


Figure 32 – mlab

After logging in, website will redirect to User main page, click “Create New” button to create database for your website. Now, it will direct to choose Cloud Provider Service (Amazon, Google Cloud Platform, Microsoft Azure) and which Plan Type User want their website to have. Amazon and Sandbox type are the greatest choice for my application to work. Then go to next step by pressing “Continue”.

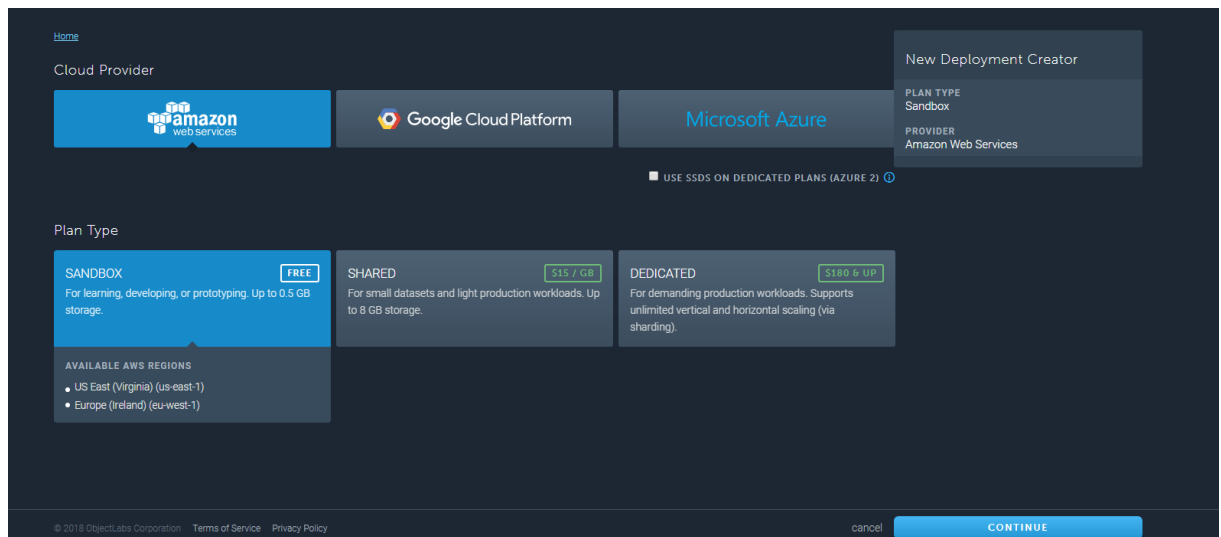
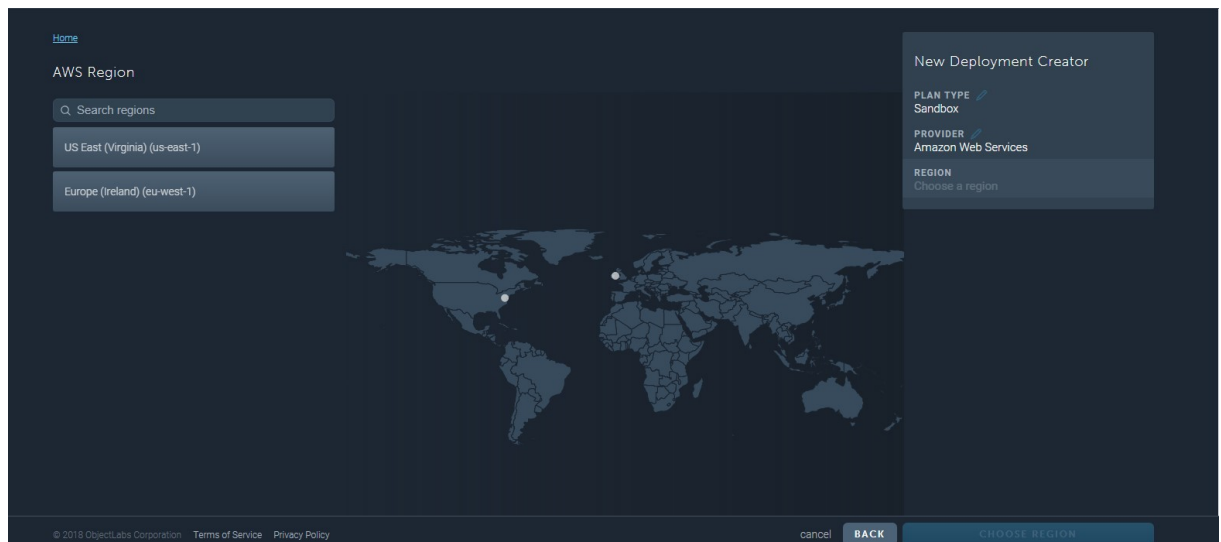


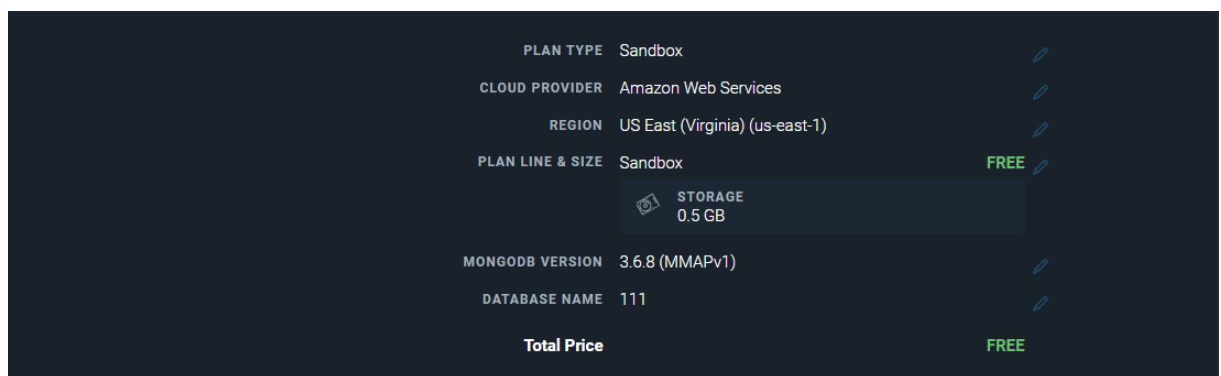
Figure 33 – mlab Provider

Choose region is also important step because it will determine the connection between user’s application and database server.



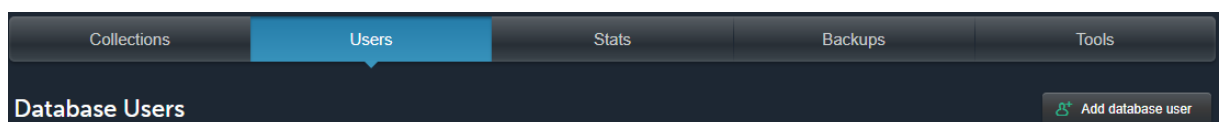
*Figure 34 – mlab Region*

Next step is to Name User database and confirm order with mlab website. As Figure 35, mlab is supporting MongoDB with MongoDB Version in the confirmation order step.



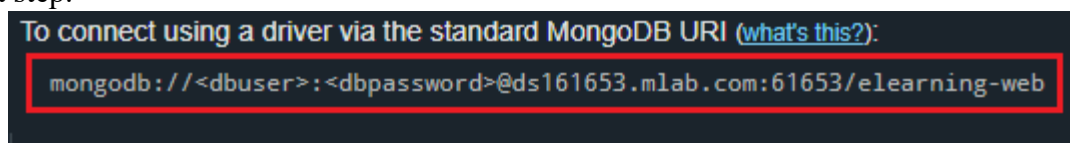
*Figure 35 – mlab Confirm*

Last step is creating a “Users” for your database, this will generate a URL for programmer to connect with mlab through NodeJS application. Access to your new created database and choose “Users” tab. And click “Add database user”, this step is to control who can connect to this database by create Name and Password for your database.



*Figure 36 – mlab Interface*

Then copy this link and change <dbuser>, <dbpassword> which was just created in the last step.



*Figure 37 – mlab connect API*

### 4.3. Results

After researching and testing of NodeJS language, the result of communicating directly from client and server is very fast as well as how user experience the website. There are many techniques that have been applied and tested to reach the final goal. This section will show 4 main features which existed on E-learning application.

#### 4.3.1. Create “room” with SocketIO.

Figure 38 shows main teacher interface. Each teacher with different id will have different assigned course (Course created by admin, then assign to teacher).

List Of Course				
Course Name	Semester	Name	Class List	Start Lesson
<u>Introduction to Web Application</u>	Semester1	teacher001	<a href="#">View</a>	<a href="#">Start</a>
<u>Computer Graphics</u>	Semester1	teacher001	<a href="#">View</a>	<a href="#">Start</a>

Figure 38 – List Of Course

Each course will also have their unique id and when teacher press “Start” button, that id will be sent to Teacher Controller to handle teacher request (For example, teacher A click Start for course “Computer Graphics” then based on “Computer Graphics” id in database, it will be sent through a href link in “a tag” like in Figure 39).

```
<td><a id="linkroom" href="/teacher/liveStream/<%= data1[i]._id; %>" class="btn1 btn1--color2" value="<%= data[i]._id %>">Start</a></td>
```

Figure 39 – Example Creating room

```
socket.on('connect', function() {
  socket.emit('creat-room', $('#linkroom').val());
});
```

Figure 40 – Emit Event create room

To send information to back-end to operate create new room, client’s javascript file will get the id of html “a tag” and send to server to handle. In Figure 40, Client will emit an event call “create-room” with value is value of “#linkroom” value (here is the course id).

Until now, client have finished their job. Event “create-room” is waiting for server to handle.

```
io.on("connection", function(socket) {
  console.log("Someone is Connected " + socket.id);

  socket.on("creat-room", function(data) {
    socket.join(data);
    console.log(data);
  });
});
```

Figure 41 – Server listen to Event

On first line of Figure 41, “io.on” means server will listen to every action which client do on website. Second line is to show the id of connection, each connection will also have their own connect id (when ever a user access to website, server will create a socket id for that user). Third line of block code show how server handle event “create-room” which sent by client.

- “Socket.on” mean server is listening to event “create-room” sent from client side with data (data here is the course id).
- “Socket.join(data)” is to subscribe a user to a given channel (In this example, teacher A will be subscribed on channel with course id).

All the above step is to create a unique room, each room will handle their own function (For example, if user A chat on channel A, all user from channel A can see that message but in channel B nothing will happen).

#### 4.3.2. Live Stream (1-many connection).

After creating room in “Section 4.3.1” above, teacher interface will be redirected to livestream page (Red box in Figure show Teacher’s camera).

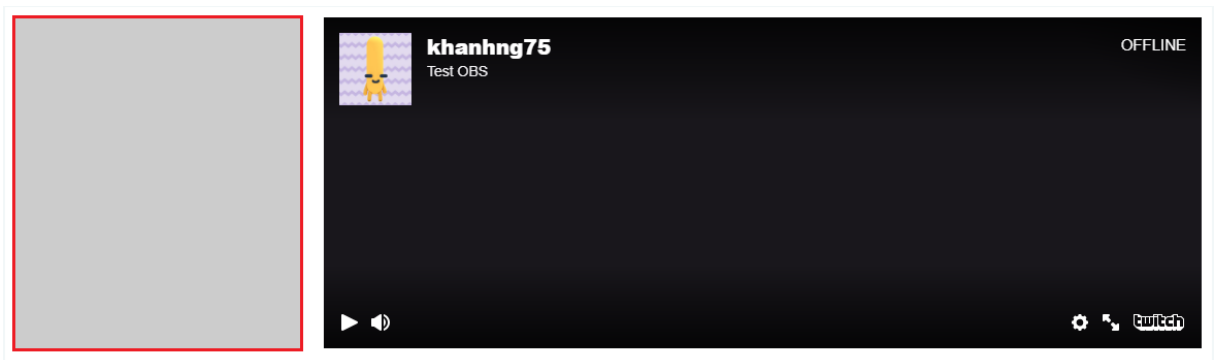


Figure 42 – Teacher Live Page

When teacher access to live stream page, it will automatically emit event “stream” for server to handle (in viewVideo function red line).

```
function viewVideo(){
    context.drawImage(video,0,0,context.width,context.height);
    socket.emit('stream',canvas.toDataURL('image/webp'));
}

$(function(){
    navigator.getUserMedia = (navigator.getUserMedia ||
    navigator.webkitGetUserMedia ||
    navigator.mozGetUserMedia ||
    navigator.msGetUserMedia );
    if(navigator.getUserMedia){
        navigator.getUserMedia({video:true,audio:true},loadCam,loadFail);
    }
    setInterval(function(){
        viewVideo(video,context);
    },70);
});
```

Figure 43 – Connect User Camera

Teacher live stream only use create “room” function of SocketIO and different technique is used for capturing Teacher’s camera and live to Student page. WebRTC (Web Real Time Communication) is also a standard which will allow Web browser to perform real-time multimedia communications such as voice, video).

In Figure 43, yellow box shown how to use WebRTC to connect with computer, laptop camera and send to User Interface. The Yellow line code will allow browser to access user camera, microphone to capture data. Programmer can set different constraints by applying new code to **getUserMedia({<new constraints>})**.

```
socket.on('stream', function(image) {
  socket.broadcast.emit('stream', image);
});
```

*Figure 44 – Server emit back Event Stream*

After sending event “stream” to sever, server will response by listening event “stream” with data “image” from teacher page and then emit event “stream” back to Student page to listen. (broadcast.emit means sending a message to everyone else except for the socket that starts it, no transfer back to teacher page but transfer the event to all client that listen to “stream” event sent from server).

```
socket.on('stream', function(image) {
  var img = document.getElementById("play");
  img.src = image;
});
```

*Figure 45 – Listen data from server*

Figure 45 shows the listening event from student page to get data “image” from server sent to and then attach the value to img.src (img here is defined as <iframe src= “ “ >). This javascript code will automatically replace the empty in iframe src with data from server. Teacher camera now is turn on and Student can watch it from their own computer.

### 4.3.3. Screen capture API.

In the Figure 42, right hand side is a iframe tag with src is the API from twitch account channel. When ever Teacher live on twitch, it will recapture and send to where ever the iframe is setting up.

An E-learning application is still in development stage so there will be some functions that can not be done by hand so getting API is the best solution for that function to work. In this function, teacher will connect their OBS (Open Broadcaster Software) with twitch channel id (Each teacher will have their own channel id in teacher profile).

```
<iframe src="https://player.twitch.tv/?channel=<teacher-channel-id>" frameborder="0" allowfullscreen="true" scrolling="no" |frameborder="0"></iframe>
```

*Figure 46 – Twitch API*

There are some replacements for OBS such as Xsplit as well as Twitch platform like Youtube, Facebook, etc. But OBS and Twitch platform is the most suitable for this project to work (OBS is an open-source that can handle almost all functions teacher need and Twitch is the platform that allow to embed Twitch API to another website).



#### 4.3.4. Firebase Connection.

The main reason for using Firebase storage is to save User Image and return image URL to mlab to save to user data (mlab does not allow to save special data like image, word file, docx, etc). After creating an account on Firebase, each User will unique API to connect with Firebase on their application.

- Connect to Firebase.

Search for the Config file in Firebase console which will have the same structure as Figure

```
<script src="https://www.gstatic.com/firebasejs/5.5.9/firebase.js"></script>
<script>
  // Initialize Firebase
  // TODO: Replace with your project's customized code snippet
  var config = {
    apiKey: "<API_KEY>",
    authDomain: "<PROJECT_ID>.firebaseapp.com",
    databaseURL: "https://<DATABASE_NAME>.firebaseio.com",
    projectId: "<PROJECT_ID>",
    storageBucket: "<BUCKET>.appspot.com",
    messagingSenderId: "<SENDER_ID>",
  };
  firebase.initializeApp(config);
</script>
```

Figure 47 – Firebase Config template

- Get Firebase API.

```
var fileButton = document.getElementById('fileButton');
if (fileButton) {
  fileButton.addEventListener('change', function(e) {
    var file = e.target.files[0];
    var storageRef = firebase.storage().ref('UserImage/' + file.name);
    var task = storageRef.put(file);
    task.on('state_changed',
      function() {
        storageRef.getDownloadURL().then(function(linkUrl) {
          console.log("File available at", linkUrl);
          var img = document.getElementById('myimg');
          var valImg = document.getElementById('myVal');
          valImg.value = linkUrl;
          img.src = linkUrl;
          downloadURL = linkUrl;
        });
      });
  }, false);
}
```

Figure 48 – Get Download URL

In Figure 48, it shows the way to obtain Download URL from Firebase storage. First, check whether input filed with id fileButton is exist or not. If exists, then when there is some change happen to this input, line 3 of Figure 48 code will start. It will get the file from User select then definid which folder to save in Firebase and append the file to variable called “task”. After connecting to saving folder, red link in this Figure will get the URL and return with linkURL. Append “linkURL” to iframe src to get the output of file (such as image, link, text, document, etc)

#### 4.3.5. Real time chat application.

Real time chat is also an important feature for creating an Online Learning Platform which will have a big impact on User Experience. Having this will help User to stay connected with Teacher whenever they have a question or misunderstand about some point in Teacher lesson.

To build this function, it requires the same logic live stream event in **Section 4.3.2** and **Section 4.3.1**. First thing is to create a unique room which will have their own workspace. Secondly, Client side emit an event “user-chat” for server side to handle and Server side will listen to this event and emit back the data sent by client.

For example, client A emit event “user-chat” with data send is “Hello everyone” like in Figure 49 (Data will get from input type text in html page)

```
$(document).ready(function() {  
    $("#btnChat").click(function() {  
        socket.emit("user-chat", $("#txtMessage").val());  
    });  
});
```

Figure 49 – Emit User-chat event

After receiving event from Client-side, Server will return data to a channel which was created before sending chat message with client A data.

```
socket.on("creat-room", function(data) {  
    socket.join(data);  
    socket.channel = data;  
});  
  
socket.on("user-chat", function(data) {  
    io.sockets.in(socket.channel).emit("server-chat", data);  
});
```

Figure 50 – Emit Data to channel

First, server-side will listen to event “user-chat” and then return data in red line code in Figure 50. This line will only send data to a specific channel, not all channel so workspace of each room will be conserved.

Every time client-side press/enter message, they are creating an event for User to listen with data from input type text and then server-side will listen to that event and emit back data to all client in the same channel.



## CHAPTER 5

### CONCLUSION AND FUTURE WORK

#### 5.1. Conclusion

The main objective of this project is to create an e-learning application that can help students in their study. With the support of socket as well as some package in NodeJS, I can create a live streaming for teacher to teach the lesson directly to student without using another third software. In this project, I was able to create some simple function like login, logout, register course, etc. There are lots of problems to considered during the development later but in this project, the main thing is to design a JSON database that can control the whole system, JSON database can only link to another table by using id field which is created by MongoDB. This system is not difficult to build, this is a more advance system compare to the project in web application subject, so it takes more time to research and test whether it work or not in the real life.

#### 5.2. Future Work

In the future, there are more functions to be complete in order to fulfill the requirement of an e-learning website. First, I will improve the UI of the website by using AngularJS rather than only using basic html, css and include Bootstrap so that it can become a responsive website which is a basic requirement of web application nowadays.

And finally, the code will be optimized in near future to increase the speed of the system and the data transfer is real time transfer without reloading page. At present, there are not much time to combine all lack of requirement, but it will be completed in near future.

## REFERENCES

- [1]: JavaScript stream.  
<https://www.sitepoint.com/stream-your-webcam-to-a-browser-in-javascript/>
- [2]: Media Stream API.  
[https://developer.mozilla.org/en-US/docs/Web/API/Media\\_Streams\\_API/Constraints](https://developer.mozilla.org/en-US/docs/Web/API/Media_Streams_API/Constraints)
- [3]: Live Stream Website.  
<https://www.romexsoft.com/blog/live-streaming-website/>
- [4]: How to create live stream platform.  
<https://www.wowza.com/community/questions/6439/a-dummys-guide-to-creating-a-live-streaming-platfo.html>
- [5]: Kurento in NodeJS  
<https://groups.google.com/forum/#!topic/kurento/yXnWUIJrnEM>
- [6]: WebRTC in NodeJS  
<https://stackoverflow.com/questions/21402840/webrtc-for-one-to-many-broadcast-using-node-js>
- [7]: NodeJS Documents  
<https://nodejs.org/en/docs/>
- [8]: MongoDB Documents  
<https://docs.mongodb.com>
- [9]: SocketIO Documents  
<https://socket.io/docs/>  
<https://viblo.asia/p/buoc-dau-lam-quen-voi-nodejs-va-socketio-MJyGjQrWvPB>