

# Acknowledgment

We wish to acknowledge with gratitude, the insightful guidance given by our module leader Dr. Rasika Ranaweerage. You have been our guidance mentor. We have been extremely lucky to have a lecturer who cared so much about student work, and who responded to our questions so promptly regarding. Not only module contents whenever he taught many extra trends, but technologies regarding our module since the second year and it also increase our knowledge.

# Table of Contents

Table of Contents

[Acknowledgment 1](#_Toc38548403)

[Table of Contents 1](#_Toc38548404)

[Table of figures 3](#_Toc38548405)

[1.0 Project identification 4](#_Toc38548406)

[1.1 Introduction 4](#_Toc38548407)

[1.2 Project Goals 4](#_Toc38548408)

[1.3 Project Objectives 4](#_Toc38548409)

[2.0 Planning 5](#_Toc38548410)

[3.0 Analyze 5](#_Toc38548411)

[3.1 Functionalities 5](#_Toc38548412)

[3.2 Requirement Analysis 6](#_Toc38548413)

[3.2.1 Functional Requirement 6](#_Toc38548414)

[3.2.2 Non-Functional Requirements 6](#_Toc38548415)

[4.0 Designing 7](#_Toc38548416)

[4.1 Architecture Diagram 7](#_Toc38548417)

[4.2 Use Case Diagram 8](#_Toc38548418)

[4.3 Database Design 9](#_Toc38548419)

[4.4.1. Extended Entity Relationship Diagram 9](#_Toc38548420)

[4.4.2. Relational Mapping 10](#_Toc38548421)

[4.4.3. Normalized Tables 11](#_Toc38548422)

[4.4.4. Database Diagram 12](#_Toc38548423)

[5.0 Implementation 13](#_Toc38548424)

[5.1 Web App Implementation 13](#_Toc38548425)

[5.1.1 Features of Web Application 13](#_Toc38548426)

[5.1.2 Web Application Sample Screenshots 14](#_Toc38548427)

[5.2 Mobile App Implementation 16](#_Toc38548428)

[5.2.1 Features of Mobile Application 16](#_Toc38548429)

[5.2.2 Mobile Application Sample Screenshots 17](#_Toc38548430)

[5.3 API Implementation 18](#_Toc38548431)

[5.3.1 Overview 18](#_Toc38548432)

[5.3.2 API Project Dependencies 20](#_Toc38548433)

[5.3.3 Model 20](#_Toc38548434)

[5.3.4 Controller 22](#_Toc38548435)

[5.3.5 Routes 23](#_Toc38548436)

[5.3.6 Configuration 26](#_Toc38548437)

[6.0 Tools and Technologies 26](#_Toc38548438)

[5.1 Technology Used 26](#_Toc38548439)

[5.2 Why Did We Use Them? 28](#_Toc38548440)

[5.3 API Development Technologies 30](#_Toc38548441)

[7.0 API Documentation 30](#_Toc38548442)

[8.0 Instruction to Run 30](#_Toc38548443)

[9.0 Contribution 31](#_Toc38548444)

[10.1 Workload Matrix 31](#_Toc38548445)

[10.2 LinkedIn Certificates 34](#_Toc38548446)

[11.0 Turnitin Report 40](#_Toc38548447)

[12.0 Google Driver Link of the Project 41](#_Toc38548448)

[13.0 References 42](#_Toc38548449)

# Table of figures

[Figure I -Architecture Diagram 7](#_Toc38543420)

[Figure II- Use Case Diagram 8](#_Toc38543421)

[Figure III- EER 9](#_Toc38543422)

[Figure IV - Relational Mapping 10](#_Toc38543423)

[Figure V - Normalized Tables 11](#_Toc38543424)

[Figure VI - Database Diagram 12](#_Toc38543425)

[Figure XI - Approved Students 14](#_Toc38543426)

[Figure XII - Non-Approved students 15](#_Toc38543427)

[Figure XIII - Rejected Students 15](#_Toc38543428)

[Figure XVII - Student Page 16](#_Toc38543429)

[Figure XXV - Login Screen 17](#_Toc38543430)

[Figure XXVI - Signup Screen 17](#_Toc38543431)

[Figure XXXIX- Project Structure 18](#_Toc38543432)

[Figure XL- server.js 19](#_Toc38543433)

[Figure XLI- package.json 20](#_Toc38543434)

[Figure XLII- Models 20](#_Toc38543435)

[Figure XLIII – db.js 21](#_Toc38543436)

[Figure XLIV - A part of the user.model.js 22](#_Toc38543437)

[Figure XLV – Controllers 23](#_Toc38543438)

[Figure XLVI- a part of the user.controller.js 23](#_Toc38543439)

[Figure XLVII-Routes 24](#_Toc38543440)

[Figure XLVIII -admin.routes.js 24](#_Toc38543441)

[Figure XLIX - expert.routes.js 24](#_Toc38543442)

[Figure L - login.routes.js 25](#_Toc38543443)

[Figure LI - resume.route.js 25](#_Toc38543444)

[Figure LII - student.route.js 26](#_Toc38543445)

[Figure LIII- user.routes.js 26](#_Toc38543446)

[Figure LIV- Database Configuration 26](#_Toc38543447)

# Project identification

## Introduction

University to Industry is a platform that allows the industry personals to directly recruit university students. The platform allows a registered university student to maintain his profile and curriculum vitae which are made visible for the registered industry personals. The industry professionals can find out the students with qualifications and skills that they are looking for using the platform.

This project is mainly about the development of the API for this system. The REST API has been developed for the CRUD operations of the database that is being used for this system. The system is also consisted of a web application as the front end that uses the API to communicate with the database.

## Project Goals

The main goal of this project is to develop a RESTful API web application and complete our project according to the given assignment criteria.

Learn something new and improve our technical skills and knowledge were our second goal as a team.

## Project Objectives

* Carefully analyses the scenario and deliver a good effective solution.
* Figure out new developing tools and technologies which can fulfill the project requirements.
* Design and development of a functional web application and a mobile application.
* The source code should have been tested properly by using test data.
* The design clearly illustrated within the website documentation and clearly evident in the architecture.
* Proper project documentation.

# Planning

Here, we identified and defined the purpose and project plan for the system development. There is the main task we have completed in this stage.

That is Requirement Gathering. Requirements gathering is an essential part of any project and project management. Understanding fully what a project will deliver is critical to its success.

Requirement gathering was mainly done by analyzing project criteria and observing the given scenario.

# Analyze

## Functionalities

From the IPT program manager(Admin) aspect,

* IPT program manager can access the websites admi panel by using username and password.
* IPT program manager can see details about any registered member.
* According to the given details IPT program manager can approve or reject any membership.
* IPT program manager can modify their account details and logout from the account.

From the student aspect,

* Any student can sign up for the website by giving required details.
* They can sign in using username and password.
* Student can edit their profile details and share their qualifications.

From the IT expert aspect,

* Any IT expert can sign up for the website by giving required details.
* They can sign in using username and password.
* Approved experts can filter by the category to find appropriate students and communicate with them.
* They can also can filter by the category and see the previous selected students and their contact details.

## Requirement Analysis

### Functional Requirement

1. Students can register themselves with details including a student ID, name, profession, email, affiliated university, password, etc.
2. Experts can register themselves with details include national ID, name, profession, email, affiliated company, password, etc.
3. Students or experts can update their profiles.
4. Anyone who logged in can access member details when the id is known.
5. An expert should be able to filter the students by category.
6. The IPT manager can remove invalid members, students or experts

### Non-Functional Requirements

* 1. Availability- The system is available for accessing anytime with the maximum number of users
  2. Reliability- The system is error-free and work according to the specifications mentioned.
  3. Security- A third party cannot access the system without any authorization from the administration.
  4. Accuracy- The system depends on real time information and the system provides real time updated data to the user. If any modification happened that will be updated in the system instantly.
  5. Maintainability- The system can be easily maintained.

# Designing

## Architecture Diagram

A screenshot of a cell phone

Description automatically generated

Figure -Architecture Diagram

## Use Case Diagram

A picture containing text, map

Description automatically generated

Figure - Use Case Diagram

## Database Design

### Extended Entity Relationship Diagram

A picture containing text, map

Description automatically generated

Figure - EER

### Relational Mapping

A screenshot of a cell phone

Description automatically generated

Figure - Relational Mapping

### Normalized Tables

A screenshot of a cell phone

Description automatically generated

Figure - Normalized Tables

### Database Diagram

A screenshot of a cell phone

Description automatically generated

Figure - Database Diagram

# Implementation

## Web App Implementation

### Features of Web Application

* Common login Page

All the users can log in to the application using this common login. They can select their user type (Admin, Industry experts, Students) and enter their other credentials(Username, Relevant ID number, Password).

* Common Signup Page

Users can register to the system throughout this page.

* Admin Panel

Following tabs are available on Admin panel

* Students
* Approved
* Non- Approved
* Rejected
* IT Experts
* Approved
* Non- Approved
* Rejected

All the newly registered students and industry experts will display through this page under the Non Approved tab. Admin can approve or reject the registered users(Industry experts, Students).

All the approved users will display under the relevant "Approved"(both Experts and Students have separate tabs) tab and all the rejected users display under the relevant "Rejected" tab.

* Students Page

After successful log in, students will direct to their account. Using this account, once student come to this page it notifies whether that student is approved , pending or rejected student. Students can insert or update their details and qualifications. Also rejected students can send request to admin for approval.

* IT Experts Page

After logged in, experts will direct to their account page. If they got the approved from the admin, they could view the approved student's details and select them. After selecting, they can view the selected students separately according to their qualification category.

### Web Application Sample Screenshots

A screenshot of a computer screen

Description automatically generated

Figure - Approved Students

A screenshot of a computer screen

Description automatically generated

Figure - Non-Approved students

A screenshot of a computer screen

Description automatically generated

Figure - Rejected Students

A screenshot of a cell phone

Description automatically generated

Figure - Student Page

## Mobile App Implementation

### Features of Mobile Application

All the features in web application can be seen in mobile application expect Admin panel.

### Mobile Application Sample Screenshots

A picture containing computer

Description automatically generated

Figure - Login Screen

A picture containing screenshot, monitor, remote, screen

Description automatically generated

Figure - Signup Screen

## API Implementation

### Overview

Apart from node modules in the backend **API** folder is heart of this application. Because it contains Restful Web API.



Figure - Project Structure

In server.js file contains configurations of server. It has whole responsible of routing and same time working as server.



Figure - server.js

### API Project Dependencies

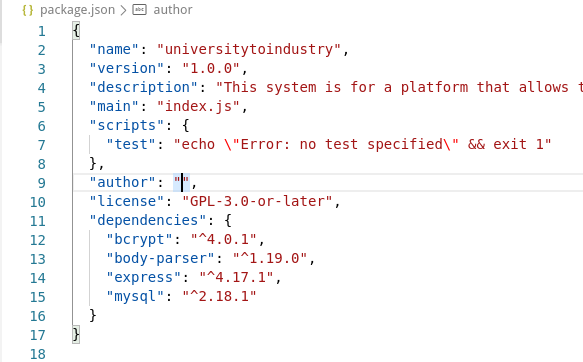


Figure - package.json

### Model

First, we designed EER and summarize table meta details as in normalized table section.

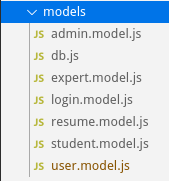


Figure - Models

Also, in the backend folder it has db.js file which is store connection string to the database.

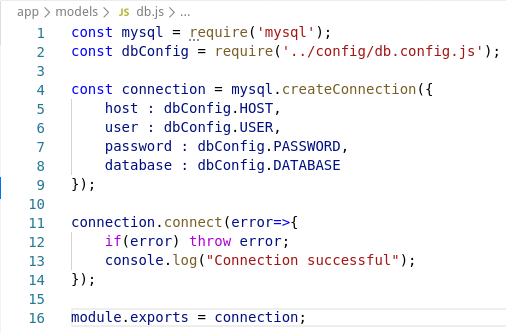


Figure – db.js

For an example we added a part of the User model.



Figure - A part of the user.model.js

### Controller

After implementing models then we implemented controllers for models. In controllers we import models and ad methods which are GET, POST, PUT, DELETE HTTP methods.

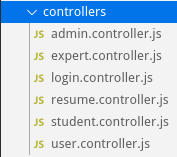


Figure – Controllers

For example we added a part of the user controller that contains the controller of just record creation.



Figure - a part of the user.controller.js

### Routes

Routes are defining the URI of particular resources along with certain HTTP methods.

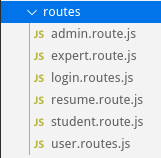


Figure -Routes



Figure -admin.routes.js



Figure - expert.routes.js

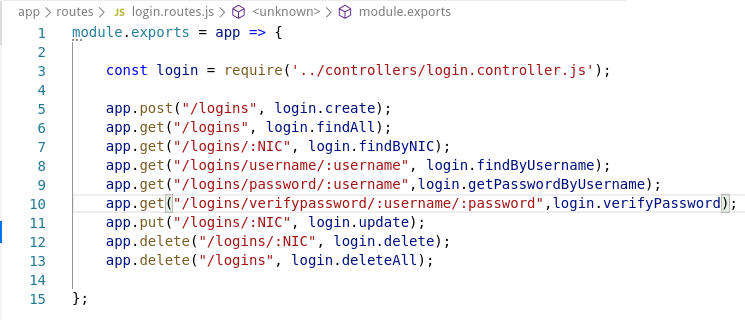


Figure - login.routes.js

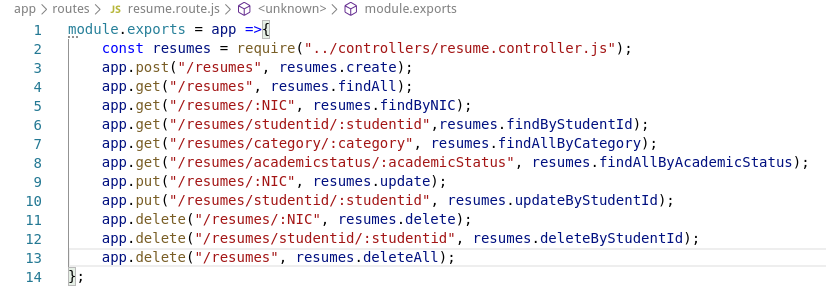


Figure - resume.route.js

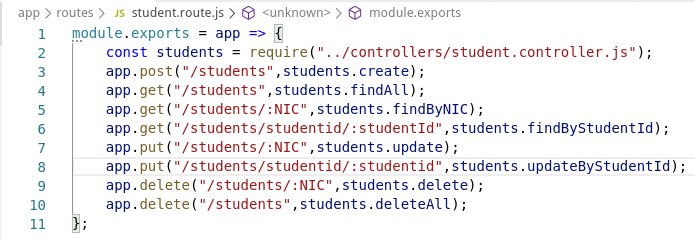


Figure - student.route.js

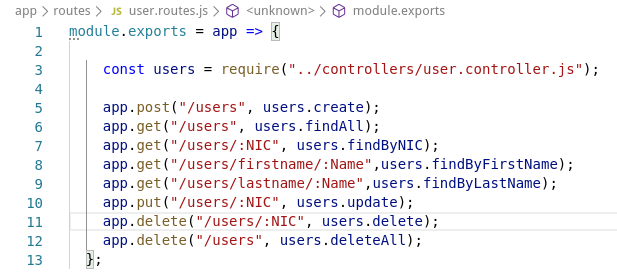


Figure - user.routes.js

### Configuration

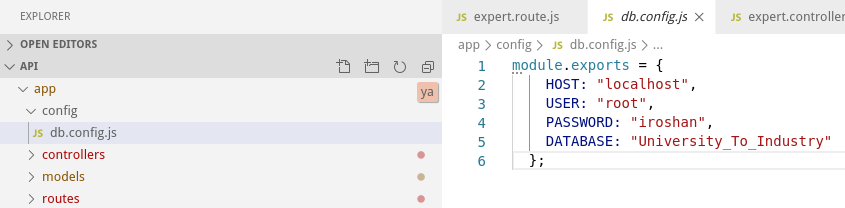


Figure - Database Configuration

# Tools and Technologies

## Technology Used

To develop this project basically, we used the following technologies.

For the backend implementation

* NodeJS

“Node.js is an open-source, cross-platform, javascript runtime environment that executes Javascript code outside of a web browser”[[1]](#footnote-1).

* ExpressJS

“Express.js, or simply Express, is a web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js”[[2]](#footnote-2).

It is Back-end web application framework running on top of Node.js

* MySQL.

“MySQL is an open-source relational database management system (RDBMS). MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos, and Tru64. A port of MySQL to OpenVMS also exists.”[[3]](#footnote-3)

For the frontend implementation

* Vue Framework

“Vue is an open-source Model–view–ViewModel JavaScript framework for building user interfaces and single-page applications”[[4]](#footnote-4).

Apart from above we used,

Bcrypt - which is a node.js password hashing method to hashing passwords.

Body-Parser - It simply let handle JSON objects in backend level.

JWT – generate token and authorize to only authenticated users.

## Why Did We Use Them?

* Node.js

Node.js is built on the JavaScript runtime of Chrome. The purpose is to develop scalable and fast network application.

The following are some of the key features that make Node.js the first choice of software architects.

* **Asynchronous and Event-Driven**

“All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the following API subsequent to calling it and a notification mechanism of Events of Node.js encourages the server to get a response from the past API call”[[5]](#footnote-5).

* **Fast**

“being based on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution”[[6]](#footnote-6).

* **Single-Threaded but Highly Scalable**

“Node.js uses a single-threaded model with event looping. The event mechanism causes the server to respond in a non-blocking way and makes the server profoundly versatile as instead to conventional servers which make limited threads to handle requests. Node.js uses a single-threaded program and the similar program can offer support to a much larger amount of requests than conventional servers like Apache HTTP Server”[[7]](#footnote-7).

* **No Buffering**

Node.js does not buffer data. This is one of the key features of this framework. It does not buffer in order to maintain higher level or performance. Node applications simply output lumps of data without buffering and this reduces any performance overhead that could cause due to buffering.

Express and Node Js both go combine manner in the development of backend. Those two interact with each other and shared their components to make interactions with frontends.

* Express.js

ExpressJS is a Node JS framework. It is a prebuilt framework that can be helpful in development of server side web applications. It can be used for the development of fast web applications. Express JS packs the properties of the Node JS and as a result it has inherently got great performance.

Other great characteristics are "simplicity, minimalism, flexibility and scalability"

Express JS to Node JS is basically what Bootstrap to HTML/CSS.

* MySQL

While developing the database both MongoDB and MySQL were considered. But MySQL was used as it was asked specifically in the coursework documentation and due to the relational nature of the data that is used in the system, a relational database was the ideal database system and therefore MySQL was used.

* Vue

Vue.js is a modern JavaScript framework for the development of single page web applications. It is an open source framework with MIT open source license.

Vue which permits you to put the HTML, CSS, and JavaScript inside to scope your component accurately. It is likewise valuable since you get fine aides like babel to deal with new syntax in JS like async/await.

## API Development Technologies

The API and the backend of the system is developed with NodeJS which is a JavaScript framework that can be used for the development of the back end of the system.

The system has a REST API developed using NodeJS for the CRUD operations with the database.

HTTP GET, HTTP POST, HTTP PUT, and HTTP DELETE methods are used in the REST API of this project and HTTP protocol is used for the data transmission between the system components.

# API Documentation

<https://documenter.getpostman.com/view/8511782/Szf9W7Hs>

A pdf of API documentation is attached with this project folder for more convenience.

# Instruction to Run

1. Start MySQL database service on your computer.
2. If it is a windows system, install XAMPP and click on the start button next to MySQL.
3. Then a database client program such as MySQL workbench or PhpMyAdmin can be used to import the database.
4. If it's a Linux system, enable MariaDB server by running the following command in the terminal.
5. systemctl start MariaDB
6. Then importing the database can be done by using a client program such as Deaver, MySQL workbench or phpMyAdmin.
7. After importing the database, navigate to the "API" folder of our submitted project folder.
8. Open a terminal inside and run the command
9. "npm init" to install all the dependencies.
10. Then "node server.js" to run the API.

# Contribution

## Workload Matrix

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Member | 10638366  K C A A  Iroshan | 10638387  M D S Tharindu | 10638374  T J M Siriwardhana | 10638504  L T S Dassanayake | 10638378  G H P Prabodhani | 10638431  A A A  Dulanja |
| API :   * Project Set up * Database Configuration (db.config.js)and Database Connection(db.js) * Server set up * Model, Controller and Routes programming for the database relation ‘Resume’.   + resume.model.js   + resume.controller.js   + resume.route.js |  |  |  |  |  |  |
| API :   * Model, Controller and Routes programming for the database relation ‘User’   + user.model.js   + user.controller.js   + user.routes.js |  |  |  |  |  |  |
| API :   * Model, Controller and Routes programming for the database relation ‘login’   + login.model.js   + login.controller.js   + login.routes.js * Password hashing using bcrypt. |  |  |  |  |  |  |
| API :   * Model, Controller and Routes programming for the database relation ‘Expert’   + expert.model.js   + expert.controller.js   + expert.routes.js |  |  |  |  |  |  |
| API :   * Model, Controller and Routes programming for the database relation ‘Admin’   + admin.model.js   + admin.controller.js   + admin.routes.js |  |  |  |  |  |  |
| API :   * Model, Controller and Routes programming for the database relation ‘Student’   + student.model.js   + student.controller.js   + student.routes.js |  |  |  |  |  |  |
| System Analysis and Database Design |  |  |  |  |  |  |
| Database Development   * Table creation * Constraints * Foreign Keys |  |  |  |  |  |  |
| Web Application Design |  |  |  |  |  |  |
| Web Application Development |  |  |  |  |  |  |
| Web Application Session handling |  |  |  |  |  |  |
| Web Application authentication |  |  |  |  |  |  |
| Mobile Application Design |  |  |  |  |  |  |
| Mobile App development |  |  |  |  |  |  |
| API Documentation |  |  |  |  |  |  |
| API Testing   * CRUD operations testing. * Error handling and console logs review |  |  |  |  |  |  |
| API Testing   * Passwords Hashing and Comparison using bcrypt: testing and debugging |  |  |  |  |  |  |
| Database Testing   * Constraint evaluation * CRUD operations testing * Field formats and size limits evaluation and testing. |  |  |  |  |  |  |
| Web Application Testing |  |  |  |  |  |  |
| Mobile Application Testing |  |  |  |  |  |  |
| Report Creation |  |  |  |  |  |  |
| LinkedIn Learning Certificate |  |  |  |  |  |  |

## LinkedIn Certificates

1. K C A A Iroshan 10638366

A screenshot of a cell phone

Description automatically generated

1. M D S Tharindu 10638387

A screenshot of a cell phone

Description automatically generated

1. T J M Siriwardhana 10638374



1. L T S Dassanayake 10638504



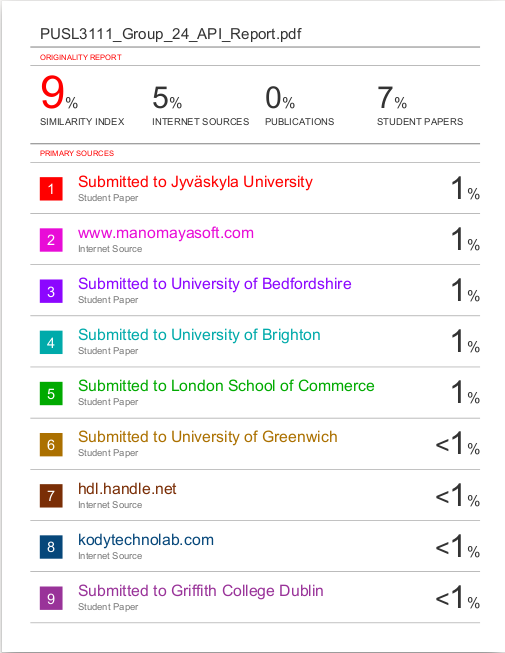
1. G H P Prabodhani 10638378



1. A A A Dulanja 10638431



# Turnitin Report



# One Driver Link of the Project

<https://liveplymouthac-my.sharepoint.com/:f:/r/personal/karunarathna_iroshan_students_plymouth_ac_uk/Documents/API%20Coursework?csf=1&web=1&e=XSMuG8>

# References

Anon., n.d. *Algoworks.* [Online]   
Available at: https://www.algoworks.com/blog/why-use-expressjs-over-nodejs-for-server-side-coding/  
[Accessed 20 04 2020].

Anon., n.d. *tutorialspoint.* [Online]   
Available at: https://www.tutorialspoint.com/nodejs/nodejs\_introduction.htm  
[Accessed 20 04 2020].

Anon., n.d. *Vue.js.* [Online]   
Available at: https://vuejs.org/v2/guide/  
[Accessed 20 04 2020].

Anon., n.d. *wikipedia.* [Online]   
Available at: https://en.wikipedia.org/wiki/Node.js  
[Accessed 19 04 2020].

Anon., n.d. *wikipedia.* [Online]   
Available at: https://en.wikipedia.org/wiki/Express.js#cite\_note-4  
[Accessed 19 04 2020].

Anon., n.d. *wikipedia-MySQL.* [Online]   
Available at: https://en.wikipedia.org/wiki/MySQL#Overview  
[Accessed 18 04 2020].

Anon., n.d. *wikipedia-Vue.* [Online]   
Available at: https://en.wikipedia.org/wiki/Vue.js  
[Accessed 18 04 2020].

1. (Anon., n.d.) [↑](#footnote-ref-1)
2. (Anon., n.d.) [↑](#footnote-ref-2)
3. (Anon., n.d.) [↑](#footnote-ref-3)
4. (Anon., n.d.) [↑](#footnote-ref-4)
5. (Anon., n.d.) [↑](#footnote-ref-5)
6. (Anon., n.d.) [↑](#footnote-ref-6)
7. (Anon., n.d.) [↑](#footnote-ref-7)