

*Tan Tao University*

School of Engineering

**Course:** Introduction to Database

## **Members**

### **Backend**

Vương Thảo Nguyên - 1902060

Lê Quang Thái - 1902097

### **Frontend**

Nguyễn Tân Thành Giang - 1805015

K'Tuýts - 1902002

**Instructor:** Dr. Cao Tien Dung

# **E-LOGBOOK WEB SERVICE REPORT**

# Table of Contents

<b>Introduction</b>	<b>3</b>
<b>Project overview</b>	<b>4</b>
Overall features	4
School administrators	4
Teachers	4
Students	4
Website's design	4
Languages, Frameworks, Libraries, and Environment	5
<b>Backend</b>	<b>9</b>
Database	9
Security	12
Deployment	17
<b>Frontend</b>	<b>19</b>
Aims and Objectives	19
Tutorial for Graphical Interface Interactions	20
Admin	20
School	20
Teacher	31
Student	36
<b>References</b>	<b>40</b>

# **Introduction**

Our project focuses on building an e-logbook system for secondary and high schools in Vietnam. The expected output is a website that supports storing not only class records but also other useful information, such as timetables or users' personal information.

The idea quickly comes into action because of the contemporary situation: most schools in Vietnam are closed due to the Covid19 pandemic [Le 2021]. Thus, the demand for having an online system for storing class records also arises. But besides the current conditions, the paper version is already hard for both storing and retrieving information. Therefore, throughout the project, we want to develop a better system when compared with the traditional management method: eLogbook website application for secondary and high school.

Libraries and Frameworks used in this project include Nodejs, express js, react js and MYSQL, Bearer authentication.

# **Project overview**

## **1. Overall features**

The e-logbook web service is designed for logbook management by a system of several schools. Here we propose an overview of functions that will be supported in this service. In detail:

- School administrators**

- + The school's administrator account will be created by the admin with username, password, and some basic background information. Under the role of the school administrator, they will be able to construct the system of their schools like creating courses, lessons, teacher's accounts, ...

- Teachers**

- + The teacher's account will be created by their specific school administrator. As a teacher, users can create and manage logbooks, manage information of students, ...

- Students**

- + Account will be created by the school administrator like the teachers' account. Student users can get timetables and check logbooks.

## **2. Website's design**

### **Server-Side Rendering (SSR) or Client Side Rendering (CSR)?**

- SSR:** With old server-side rendering solutions, you built a web page—with PHP for example—the server compiled everything, included the data and delivered a fully populated HTML page to the client. It was fast and effective. But... every time you navigated to another route, the server had to do the work all over again:

Get the PHP file, compile it, and deliver the HTML, with all the CSS and JS delaying the page load to a few hundred ms or even whole seconds.

- **CSR:** Server-side rendering is the most common method for displaying information onto the screen. It works by converting HTML files in the server into usable information for the browser. Whenever you visit a website, your browser makes a request to the server that contains the contents of the website [Vega 2017]. The request usually only takes a few milliseconds, but that ultimately depends on a multitude of factors:
  - + Your internet speed
  - + The location of the server
  - + How many users are trying to access the site
  - + How optimized the website is, to name a few

## **The team decision: CSR.**

- **Pros:**
  - Rich site interactions
  - Fast website rendering after the initial load.
  - Great for web applications.
  - Robust selection of JavaScript libraries.
- **Cons:**
  - Low SEO (Search Engine Optimization) if not implemented correctly.
  - The initial load might require more time.
  - The first load requires large space

## **3. Languages, Frameworks, Libraries, and Environment**

- **Node.js** is the greatest tool for building real-time web applications. It provides cross-platform applications which run easily on any web. So you basically don't

need anything extra for running up a node application ["Express/Node introduction - Learn web development | MDN", 2021]. You only need to make one.

### ★ Benefits of Node.js:

- + Great performance! Node was designed to optimize throughput and scalability in web applications and is a good solution for many common web-development problems.
- + Code is written in "plain old JavaScript", which means that less time is spent dealing with "context shift" between languages when you're writing both client-side and server-side code.
- + JavaScript is a relatively new programming language and benefits from improvements in language design when compared to other traditional web-server languages (e.g. Python, PHP, etc.).
- + The node package manager (NPM) provides access to hundreds of thousands of reusable packages. It also has best-in-class dependency resolution and can also be used to automate most of the build toolchain.
- + Node.js is portable. It is available on Microsoft Windows, macOS, Linux, Solaris, FreeBSD, OpenBSD, WebOS, and NonStop OS.
- + It has a very active third-party ecosystem and developer community, with lots of people who are willing to help.
- **Express** is the most popular *Node* web framework and is the underlying library for a number of other popular Node web frameworks. It provides mechanisms to:
  - + Write handlers for requests with different HTTP verbs at different URL paths (routes).
  - + Integrate with "view" rendering engines in order to generate responses by inserting data into templates.
  - + Set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response.

- + Add additional request processing "middleware" at any point within the request handling pipeline.

★ **Why use Express?** Express.js supports JavaScript which is a widely used language that is very easy to learn and is also supported everywhere. With the help of Express.js, we can easily build different kinds of web applications in a short period of time. Express.js provides simple routing for requests made by clients. It also provides a middleware that is responsible for making decisions to give the correct responses for the request made by the client

- **Javascript** is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Where HTML and CSS are languages that give structure and style to web pages, JavaScript gives web pages interactive elements that engage a user.

★ Reason to use: Javascript was created to work in browsers, and React or Node.js also use JavaScript.

- **React** is a library for building user interfaces. React is not a framework – it's not even exclusive to the web. It's used with other libraries to render to certain environments. For instance, React Native can be used to build mobile applications.

To build for the web, developers use React in tandem with ReactDOM. React and ReactDOM is often discussed in the same spaces as — and utilized to solve the same problems as — other true web development frameworks. When we refer to React as a "framework", we're working with that colloquial understanding.

## ★ **Why use React?**

- Simplicity
- Easy to learn
- Performance

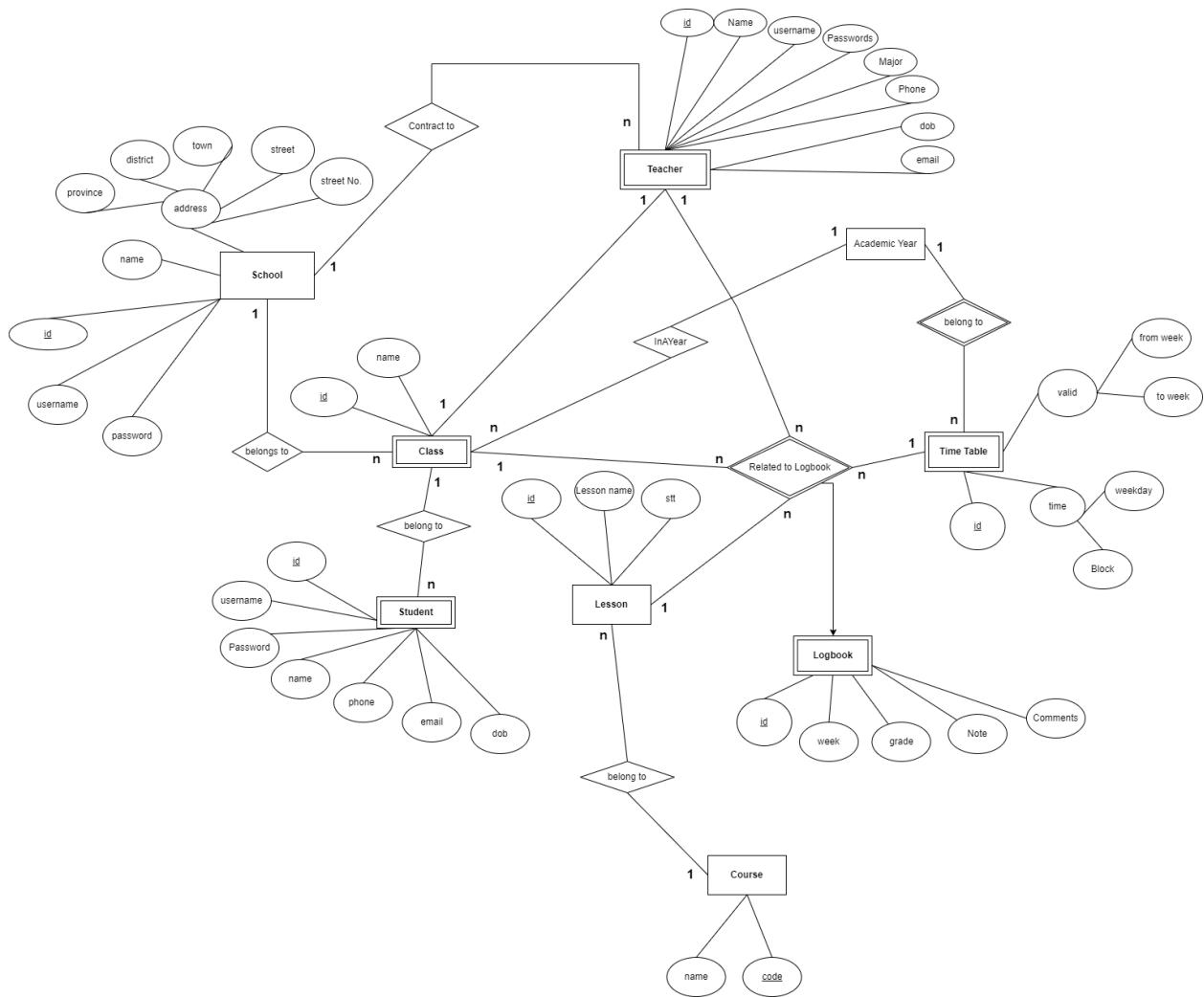
- Testability
- **ORM:** Sequelize is a promise-based Node.js ORM for Postgres, MySQL, MariaDB, SQLite, and Microsoft SQL Server. It features solid transaction support, relations, eager and lazy loading, read replication, and more.
  - ★ **Why use ORM:** Sequelize? Because our project uses MySQL to store data, it works well with ORM: Sequelize, with the help of ORM, we do not need to write raw queries → easy to read.
- **Docker** is a Linux-based, open-source containerization platform that developers use to build, run, and package applications for deployment using containers.

# Backend

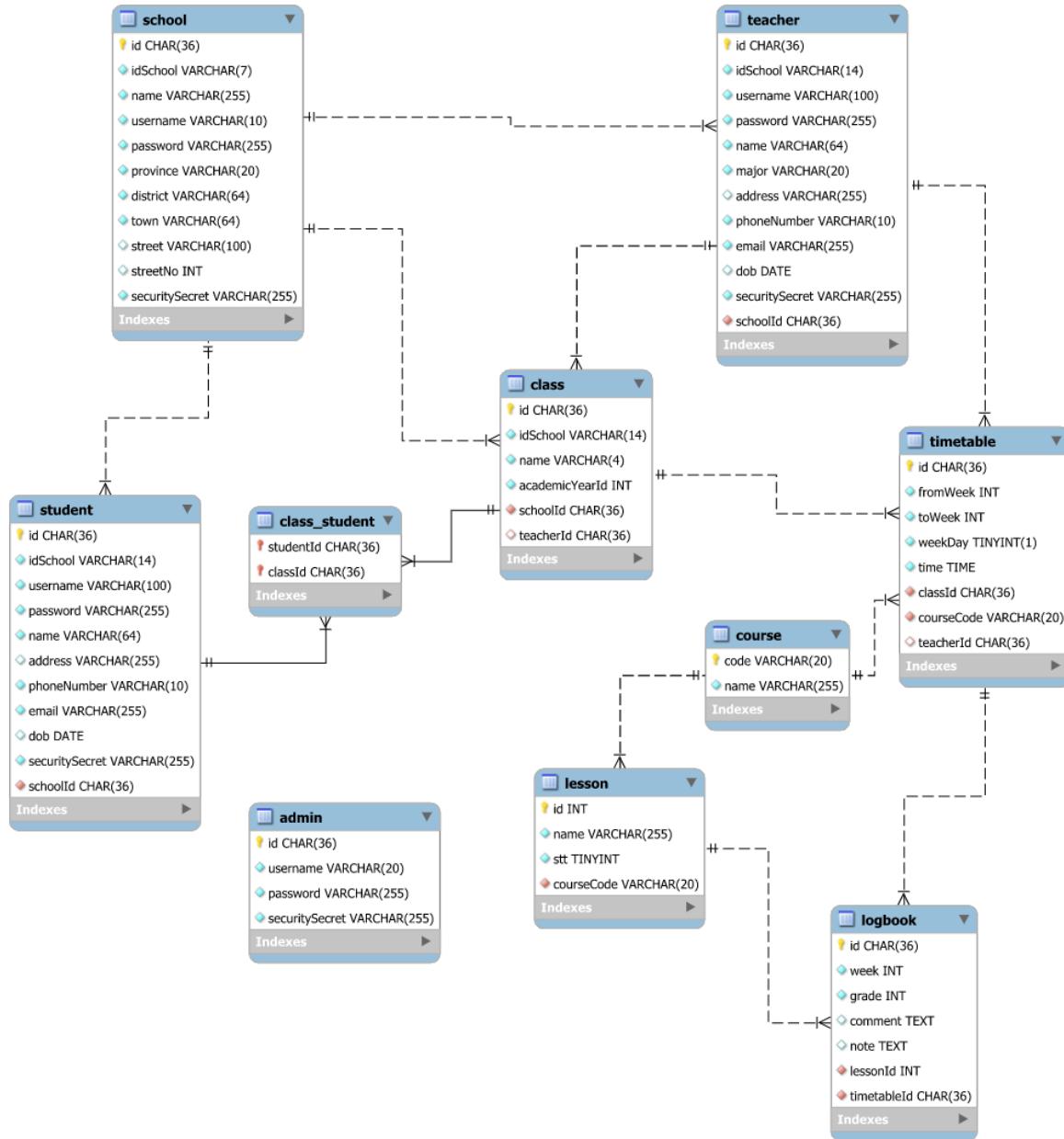
## 1. Database

### - Design:

#### - ER diagram



→ Convert it to the MySQL database



The database design satisfies **the fourth normal form (4NF)**:

- Each column must contain an atomic value.
- A column should contain values that are of the same type.
- Each column should have a unique name.
- The order in which data is saved doesn't matter.
- No partial dependencies (2NF).
- No transitive dependencies (3NF).

- For any dependency  $A \rightarrow B$ , A should be a super key (BCNF).
- No multivalued dependencies (4NF).

For the tables School, Teacher, and Student, **id**, **idSchool**, **(phone number)**, and **username** are the candidate keys (because of its uniqueness). Choosing id to be the primary key of these tables because of its nature: It's hard to guess other primary keys even when knowing the id of some users. Although knowing one's phone number is useless for guessing the other's, the field is not secure enough. The other two fields, idSchool, and username have a pattern in them. Therefore, **the most appropriate one to use as the primary key is id - which is made of a random string consisting of 36 characters.**

Users are not stored in one place but are grouped into different tables based on their roles (admin, school, teacher, student).

- **Benefits: Specification.** Fewer rows to search on for each target user. Also, don't need to use an additional field "role" to distinguish between these different target users if we group them into just one table.
- **Drawbacks:** Users are stored separately → need to know the target user to search for the information → find a way to provide this information for every request to the server → The chosen **strategy:** sending the type of user when login, then saving this piece of information in the token so that we can know the identity of the user (the table which user belong to and the id) for every next request.

### - Login

The screenshot shows a Postman API request for a 'Login' endpoint. The method is 'POST' and the URL is 'localhost:3000/auth/login'. The 'Body' tab is selected, showing a JSON payload:

```

1 {
2   "username": "logbookv1",
3   "password": "logbookv1",
4   "role": "admin"
5 }

```

The 'Body' tab has a green dot icon indicating it contains valid JSON. Other tabs shown include 'Params', 'Authorization', 'Headers (10)', 'Pre-request Script', 'Tests', and 'Settings'. The 'Body' dropdown shows options: 'none', 'form-data', 'x-www-form-urlencoded', 'raw' (selected), 'binary', 'GraphQL', and 'JSON' (selected).

- Save the type of user in the token

```
const sign = async (userId, role, signOption, userSecret) => {
  return jwt.sign(
    { sub: userId, role: role },
    signOption.SECRET + userSecret,
    { expiresIn: signOption.EXPIRED, }
  );
};
```

- Use this information to find the specific table.

```
const getUserStrategy = {
  admin: async (id) => {
    const user = await db.admin.findOne({ where: { id } });
    return user.toJSON();
  },
  school: async (id) => {
    const user = await db.school.findOne({ where: { id } });
    return user.toJSON();
  },
  teacher: async (id) => {
    const user = await db.teacher.findOne({ where: { id } });
    return user.toJSON();
  },
  student: async (id) => {
    const user = await db.student.findOne({ where: { id } });
    return user.toJSON();
  },
};
```

## 2. Security

- **Register:** only store hashing password, not the original one: even if the database system is hacked, the user's original passwords are not stolen from the database. Many surveys come to the conclusion that a lot of internet users tend to use the same or similar password for almost all of their internet accounts [Vojinovic 2021]. Therefore, once the web user loses its password on one particular website, there is a great chance that the password for other

applications of this user may also be spoiled. It is dangerous since other applications may include banking or other paid websites.

```
const salt = randomBytes(32);
teacher.password = await argon2.hash(teacher.password, {
  salt,
});
```

- **Login: secured by web token method**, specifically implemented by **HTTP Bearer authentication**. HTTP Bearer authentication method is quite commonly used in most of the websites that choose to use tokens as their security approach [Kumar 2021].

The screenshot shows the Postman application interface. At the top, there is a header bar with 'POST' and 'localhost:3000/admin/createSchool'. Below the header, there are tabs for 'Params', 'Authorization' (which is highlighted with a red box), 'Headers (11)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. On the right side of the interface, there is a 'Send' button. Under the 'Authorization' tab, there is a 'Type' dropdown menu with options: 'Inherit auth from parent', 'No Auth', 'API Key', and 'Bearer Token' (which is also highlighted with a red box). A tooltip message above the dropdown says: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables'.

- Token is like an identity card for each user. It is an encoded string that stores some information that can be used to determine the identity of the user. That encoded string stores the id, role a secret string to verify whether it is signed by our website or not, and the expired time. After login in, it is provided and stored somewhere in the local storage in the user's devices so that he/she

doesn't need to login for every request sent to the server, for a specific period of time.

```
const sign = async (userId, role, signOption, userSecret) => {
  return jwt.sign(
    { sub: userId, role: role },
    signOption.SECRET + userSecret,
    { expiresIn: signOption.EXPIRED, }
  );
};

const signAccessToken = async (userId, role, userSecret) =>
  await sign(userId, role, config.accessToken, userSecret);

const signRefreshToken = async (userId, role, userSecret) =>
  await sign(userId, role, config.refreshToken, userSecret);
```

- signOption variable refers to either the access token or the refresh token, whose configurations are as below.

```
accessToken: {
  SECRET: "sf sdf dsf dsfewrwerlkvasldkfjfdfddsf sad",
  EXPIRED: "5m",
},
refreshToken: {
  SECRET: "fsdfdsasdfsafasdadffsafhjddfdfddd",
  EXPIRED: "5d",
},
```

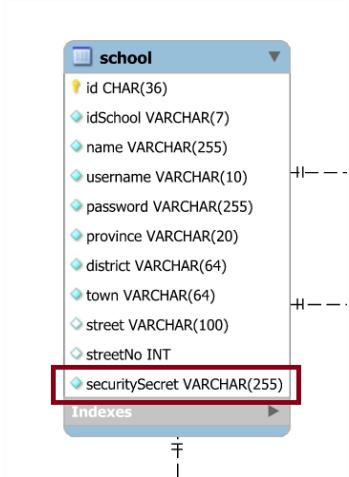
- **There are two types of tokens used in this website application**
  - **Access token:** expired every 5 minutes
  - **Refresh token:** expired every 7 days.
- Using two types of tokens is for the purpose of more security: access token is sent with a quite high frequency (for every request after login) → there is a small but possible risk that it could be stolen from hacker → setting its

validation for a short time period → even if the user loses it, there's not much time to exploit the user account.

- After the access token is expired, if users send the next request within 7 days (the validated time for refresh token), the refresh token could be used to retrieve new access and refresh token, it is done automatically by the website implementation, so that users don't need to login again to make that request. It enhances users' experience with the website: no need to login for every single call but still have some protection over the hacker from the internet. But if the web user does not send any request to the server for more than 7 days, he/she needs to login again for the next request. It also implies that if the user sends at least one request to the server a week, he/she just needs to login in once but can use the website without any disruption or further requirement.
- **Change password:** after changing the password, we need to automatically log out all the devices associated with that account. So along with the implementation of updating the user's password, we also need to provide the Log-out all device function. In other words, the task is to invalidate all the previously signed tokens, including both access and refresh ones.

→ **Chosen strategy:** create an additional field called “**securitySecret**” in all target users’ tables. securitySecret consists of 32 random characters generated by the RandomByte function and it is used to verify the token. Therefore, whenever changing the password, if the securitySecret value is also changed, all the previously signed tokens will automatically be disabled because the security key in the token no longer matches the one stored in the database when we retrieve it to verify the token.

- securitySecret field



- Sign token

```
const sign = async (userId, role, signOption, userSecret) => {
  return jwt.sign(
    { sub: userId, role: role },
    signOption.SECRET + userSecret,
    { expiresIn: signOption.EXPIRED, }
  );
};
```

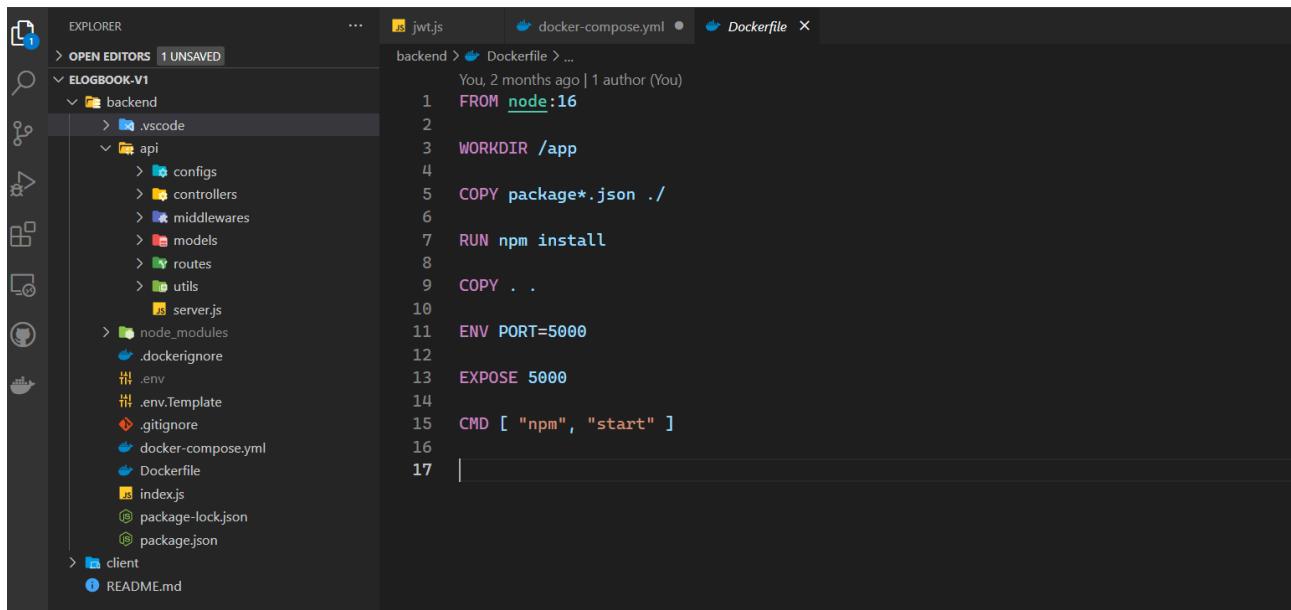
- Verify token

```
const verify = (token, verifyOption, userSecret) => {
  return new Promise((resolve, reject) => {
    return jwt.verify(
      token,
      verifyOption.SECRET + userSecret,
      (err, decoded) => {
        if (err) {
          resolve();
        }
        resolve(decoded);
      }
    );
  });
};
```

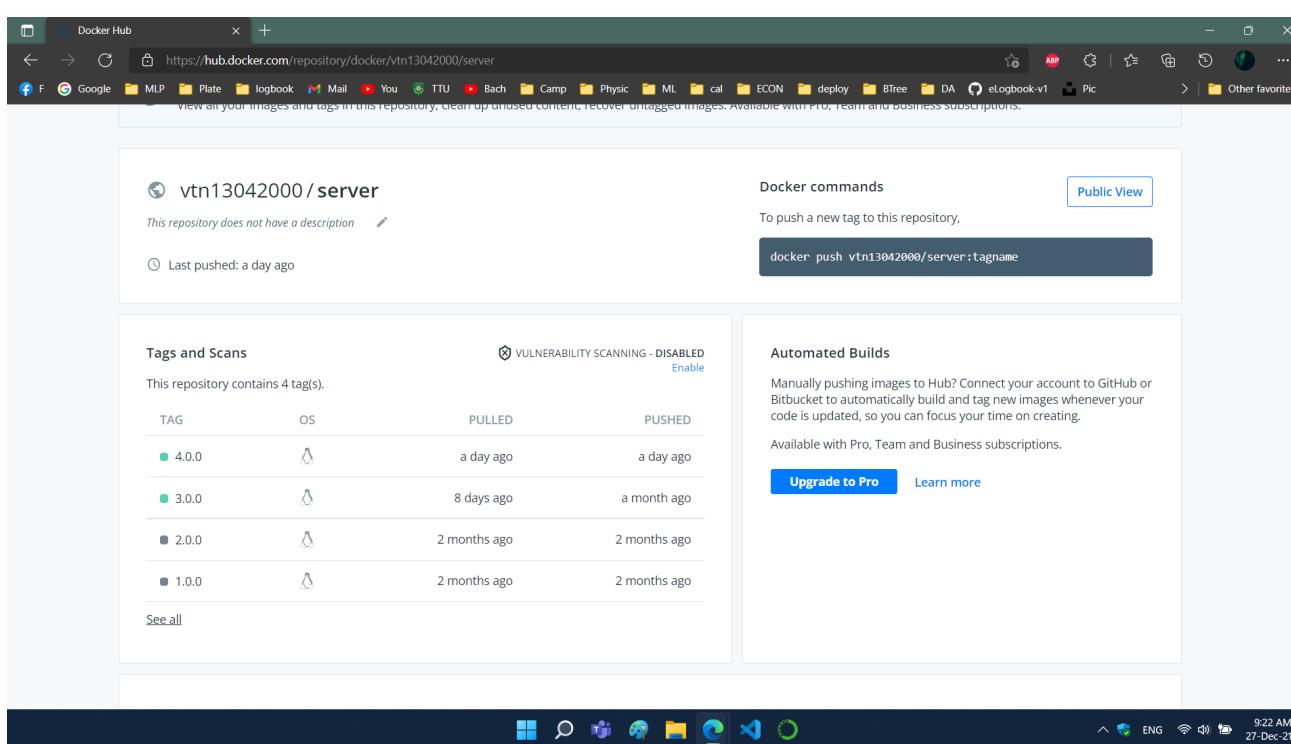
- userSecret variable refers to the securitySecret field stored in the database.
- verifyOption variable already explained in the database section.

### 3. Deployment

- Backend is deployed using Docker
  - Build the Nodejs application using the Docker file, then push it to Dockerhub.



A screenshot of a code editor (VS Code) showing a project structure for a Node.js application. The project includes an `api` folder containing `controllers`, `middlewares`, `models`, `routes`, and `utils`. The `Dockerfile` is open in the editor, defining a build process from `node:16`, setting the `WORKDIR` to `/app`, copying `package*.json` files, running `npm install`, copying the build output, setting the `PORT` environment variable to `5000`, exposing port `5000`, and running `npm start`.



A screenshot of the Docker Hub website showing the repository `vtn13042000/server`. The repository has no description and was last pushed a day ago. It contains four tags: `4.0.0`, `3.0.0`, `2.0.0`, and `1.0.0`. The `4.0.0` tag was pulled a day ago and pushed a day ago. The `3.0.0` tag was pulled 8 days ago and pushed a month ago. The `2.0.0` and `1.0.0` tags were both pulled and pushed 2 months ago. A Docker command `docker push vtn13042000/server:tagname` is shown for pushing a new tag. The page also features sections for automated builds and vulnerability scanning.

- Using Docker compose (version 3.8) to connect The Nodejs application and the MYSQL database to form the backend.

```

1 version: "3.8"
2 services:
3   mysql:
4     image: mysql:5.7
5     hostname: mysql
6     container_name: docker-logbook-mysql-4
7     restart: unless-stopped
8     environment:
9       - MYSQL_ROOT_PASSWORD=root
10      - MYSQL_USER=logbook
11      - MYSQL_PASSWORD=logbook
12      - MYSQL_DATABASE=logbook
13     volumes:
14       - db:/var/lib/mysql
15     command: --character-set-server=utf8mb4 --collation-server=utf8mb4_unicode_ci
16   server:
17     hostname: logbookserver
18     ports:
19       - 3333:8000
20     depends_on:
21       - mysql
22     image: vtn13042000/server:4.0.0
23     container_name: docker-logbook-server-4
24     restart: unless-stopped
25
26     environment:
27       - DB_HOST=mysql
28       - DB_NAME=logbook
29       - DB_USERNAME=logbook
30       - DB_PASSWORD=logbook
31       - DB_DIALECT=mysql
32       - DB_PORT=3306
33       - PORT=8000
34
35   volumes:
36     db:

```

# Frontend

## 1. Aims and Objectives

This website on the frontend side is designed to have the graphical user interface for e-logbook web service. There are several functions that were designed to serve different levels of users (super admin, school administrators, teachers, and students). The user can change their own information when it may not be precisely recorded.

In the super admin section, there is no graphical interface. It will be manipulated via API.

In the school administration section, the user can have all authority on CRUD with students, teachers, classes, and a few relationships between them which provides the information of which class that students and teachers belong to. Besides that, the courses and lessons created by the admin will be retrieved by the school administrator. Then, the timetable will be taken based on the integration of class, course, schedule, and the expiry week. Based on the timetable and respective schedules, the logbook will be constructed by the combination of timetable and specific information (teacher, lessons, comments, and notes,...).

In the teacher administration section, the user can have all authority on CRUD with logbooks as well as read some information of their own timetable.

In the student's administration, they can get the logbook information and the timetable in reading mode.

**Table 1:** *The levels of object's authority in Elogbook service:*

Objects/ User role	Admin	School	Teacher	Student
Course	CRUD	R	N	N
Lesson	CRUD	R	N	N
School	CRUD	N	N	N
Teacher	N	CRUD	N	R
Student	N	CRUD	N	R
Timetable	N	CRUD	R	R
Logbook	N	R	CRUD	R

Class	N	CRUD	R	R
-------	---	------	---	---

P/s: C: Create; R: read; U: Update; D: Delete; and N: Null.

## 2. Tutorial for Graphical Interface Interactions

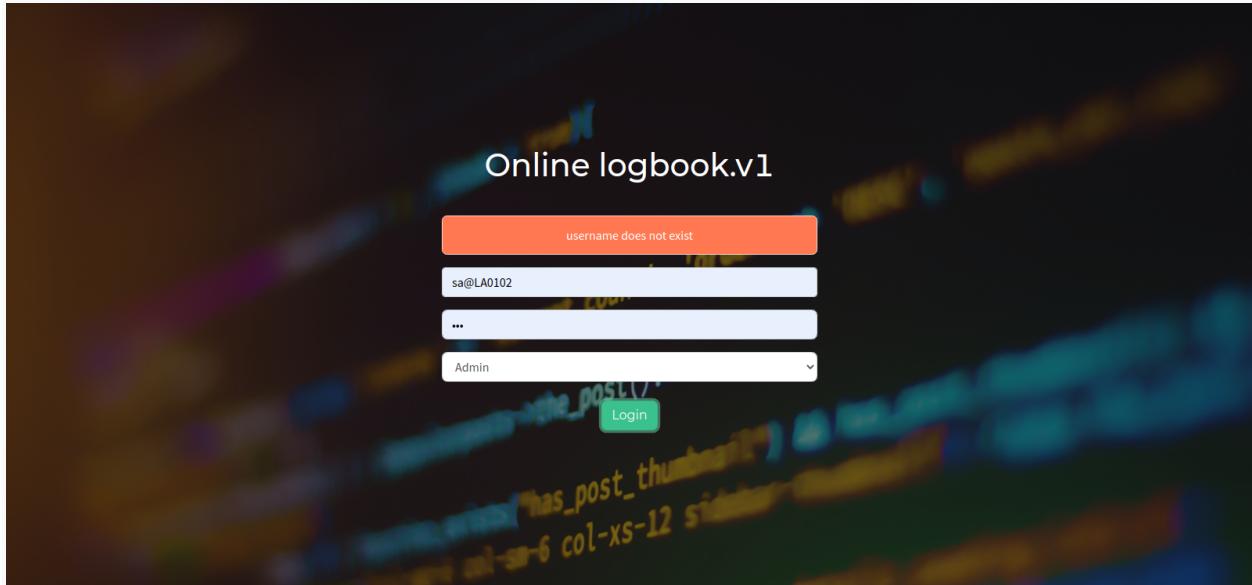
### Admin

- Admin does not have an interactive web interface, but mainly through the API that will manage the courses, lessons, and the school administrator accounts.

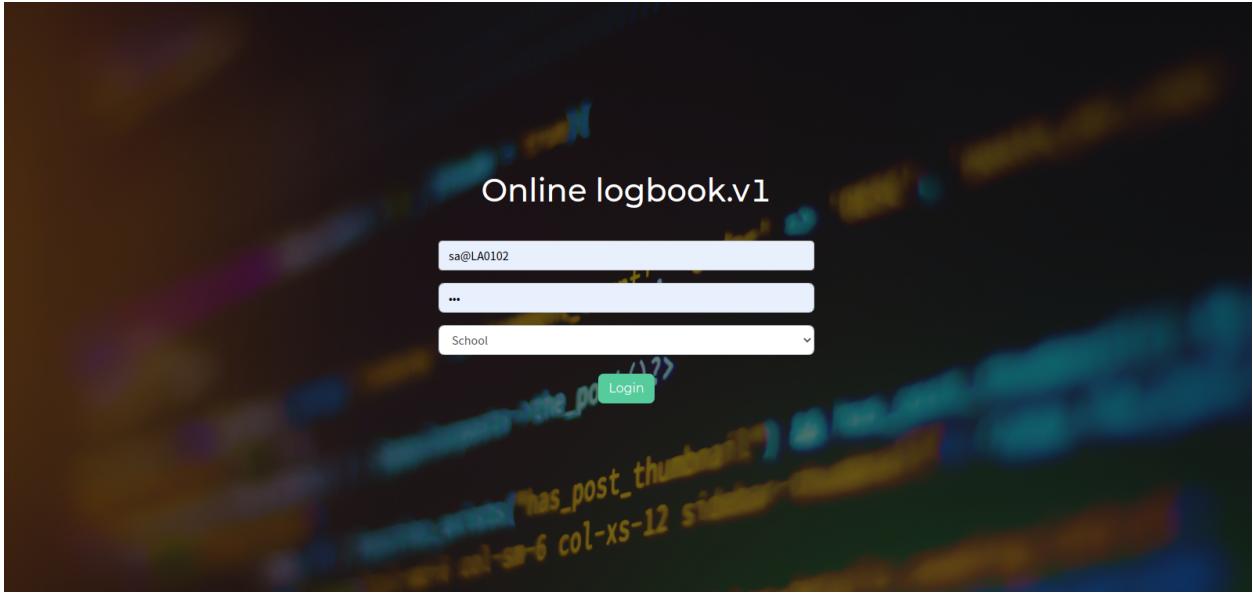
### School

#### a) Login

- The school account is provided by the admin.
- After login, it will be redirected to the homepage



- The school account is provided by the admin.
- After login, it will be redirected to the homepage



- The access and refresh tokens will be recorded in the session storage.
- If the expired time is over, the refresh token will be sent and the new tokens.

Screenshot of the Chrome DevTools Application tab showing session storage. The 'Session Storage' section for the domain 'http://localhost' is expanded, displaying the following data:

Key	Value
user	{"id": "7e90fc85-5b1c-4dc4-8c9e-de041316c57d", "username": "sa@LA0102", "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWl0IjoiZTA0MTMxNmM1N2QilCjy..."} {"rendererID": "1", "path": [{"displayName": "App:0"}, {"key": null, "index": 0}, {"displayName": "App", "key": null, "index": 0}, {"displayName": "AuthProvider", "key": null, "index": 0}, {"displayName": "Context.Provider", "key": null, "index": 0}], "role": "school", "username": "sa@LA0102"}

- After login, redirect to the homepage.

**Online Logbook.v1**

The current timetable					
Date	Time	8A7	9A9	8A1	9A2
Monday	07:30:00	NA	SBE14ECON101	NA	NA
Monday	08:30:00	NC10SINHHOC10	NA	NC18NGUVAN10	SHL11HIS101
Monday	09:30:00	NA	NA	NA	NA
Monday	10:30:00	NC18NGUVAN10	NA	NA	NA
Monday	11:30:00	NA	NA	NA	NA
Monday	13:30:00	NA	SE19CS311	NA	NA
Monday	14:30:00	NA	SHL11HIS101	NA	NA

TuytGiang Design ©2021

## b) Account information

- Update information. The response from the server will be sent via the toast.

**Online Logbook.v1**

Account Information

School Name	Tân Tạo University
Province	Long An
District	Duc Hoa
Town	Hau Nghia
Street	24A
Street Number	24

Submit

TuytGiang Design ©2021

- Change password

**Online Logbook.v1**

Update password

Old password

Type your old password here ..

New password

Type your new password here ..

Submit

TuytGiang Design ©2021

- Logout All Current Login.

**Online Logbook.v1**

Logout All

Submit

TuytGiang Design ©2021

### c) Management

- Create Students
- Example: Add one student in case of a new student who has just come.

**Online Logbook.v1**

New Student

Username	<input type="text"/>
Password	<input type="password"/>
Name	<input type="text"/>
Phone	<input type="text"/>
School	<input type="text"/>
Email	<input type="text"/>

Import File

Drop CSV file here or click to upload

**Submit**

TuytGiang Design ©2021

- Create Students
- Example: Add all students in the school using a csv file.

	A	B	C	D	E	F
1	idSchool	username	password	name	phoneNumber	email
2	LA0102-190102	nouyen.vuong.190102@LA0102	vtn	Vương Thảo Nguyễn	934565533	nouyen.vuong1902060@gmail.com
3	LA0102-190202	thai.le.190202@LA0102	vtn	Lê Quang Thái	934567833	thaleisme@gmail.com
4	LA0102-190203	ktuyt.190203@LA0102	vtn	K' Tuyt	934567833	ktuyt@gmail.com
5	LA0102-180204	giang.nouyen.180204@LA0102	vtn	Nguyễn Tân Thành Giang	934567833	nguyen.giang@gmail.com
6	LA0102-190205	hoang.dao.190205@LA0102	vtn	Đào Văn Ngọc Hoàng	4409678334	hoang.dao@gmail.com
7	LA0102-190206	tuan.tran.190206@LA0102	vtn	Trần Triều Tuấn	4409678334	tuan.tran@gmail.com

- Create Teachers
- Example: Similarly add teachers with a single form.

**Online Logbook.v1**

New Teacher

idSchool	<input type="text"/>
Username	<input type="text"/>
Password	<input type="password"/>
Name	<input type="text"/>
Phone	<input type="text"/>
Major	<input type="text"/>
Email	<input type="text"/>

Import File

Drop CSV file here or click to upload

**Submit**

TuytGiang Design ©2021

- Create Teachers
- Example: Similarly add teachers with a csv file.

	A	B	C	D	E	F	G
1	<u>idSchool</u>	<u>username</u>	<u>password</u>	<u>name</u>	<u>major</u>	<u>phoneNumber</u>	<u>email</u>
2	LA0102-T12001	dung.cao.t12001@LA0102	vtn	Cao Tiến Dũng	KHMT	987111122	dung.cao@gmail.com
3	LA0102-T13220	khanh.tran.t13220@LA0102	vtn	Trần Vũ Khanh	Toán học	934565533	khanh.tran@gmail.com
4	LA0102-T15022	hien.tran.t15022@LA0102	vtn	Trần Duy Hiển	Toán học	934567833	hien.tran@gmail.com
5	LA0102-T19222	dung.duong.t19222@LA0102	vtn	Dũng Dương	Toán	934567833	dung.duong@gmail.com
6	LA0102-T18553	fransico.roma.t18553@LA0102	vtn	Fransico Roma	Business	934567833	roma.fransico@gmail.com
7	LA0102-T12111	jonathan.lankford.t12111@LA0102	vtn	Jonathan Lankford	SHL	4409678334	jonathan.lankford@gmail.com

- Create Classes
- Example: Add all classes in the school using csv file or add a single one by the form

**Online Logbook.v1**

New Class

School id ..

Class name..

Academic year..

Import File

Drop CSV file here or click to upload

Submit

TuytGiang Design ©2021

- Create Classes
- Example: Add a single class by the form

	A	B	C
1	<u>idSchool</u>	<u>name</u>	<u>academicYearId</u>
2	LA0102-C19001	9A9	2019
3	LA0102-C19002	9A2	2019
4	LA0102-C19003	8A7	2019
5	LA0102-C19004	8A1	2019

- Add students to classes
- Example: Add students to classes

	A	B	C	D
1	<del>idSchool</del>	<del>class_name</del>	<del>Student_school_id</del>	<del>student_name</del>
2	LA0102-C1900	9A9	LA0102-190102	Vương Thảo Nguyên
3	LA0102-C1900	9A9	LA0102-190202	Lê Quang Thái
4	LA0102-C1900	9A2	LA0102-190203	K' tuyt
5	LA0102-C1900	9A2	LA0102-180204	Nguyễn Tân Thanh Giang
6	LA0102-C1900	8A7	LA0102-190205	Đào Văn Ngọc Hoàng
7	LA0102-C1900	8A7	LA0102-190206	Trần Triều Tuấn
8	LA0102-C1900	8A1	LA0102-180207	Châu Hoài Vũ
9	LA0102-C1900	8A1	LA0102-180201	Nguyễn Đức Nam

Online Logbook.v1

Notifications

Your Information

Management

Student

Teacher

Class

Create Class

Add Student

Add Teacher

Add Head Teacher

Class List

Timetable

Tutorial!

Import File

469 B  
classAddAlreadyExistStudent.csv

Fail

Submit

TuyGiang Design ©2021

- Add teachers
- Example: Add teachers to classes

	A	B	C	D
1	<del>idSchool</del>	<del>name</del>	<del>teacherId</del>	<del>teacherName</del>
2	LA0102-C1900	9A9	LA0102-T1200	Cao Tiến Dũng
3	LA0102-C1900	8A1	LA0102-T1322	Trần Vũ Khanh
4	LA0102-C1900	9A2	LA0102-T1502	Trần Duy Hiện
5	LA0102-C1900	8A1	LA0102-T1922	Dũng Dương

**Online Logbook.v1**

Add Teacher to Class

idSchool ..

Name ..

Teacher ID ..

Import File

233 B  
classAddTeacher.csv

Submit

TuytGiang Design ©2021

- Add headteachers.
- Example: Add all students, teachers, and classes

	A	B	C	D	E	F	G	H	I	J	K
1	idSchool	class_name	academicYear	teacherId	Student_school_id	classId	username	password	name	phoneNumber	email
2	LA0102-C180048AB	2018 LA0102-T15023	2018 LA0102-T15023	LA0102-180207	LA0102-C19014	yu.chau.180207@LA0102.vnn	ytn	Châu Hoài Vũ	4409678334	yu.chau@gmail.com	Nhân Dân Việt Nam

**Online Logbook.v1**

Import File

248 B  
addAlexample.csv

Submit

TuytGiang Design ©2021

- Query and get detailed information if the typo occurred, it can be deleted or updated.  
Similar in case of teacher

**Online Logbook.v1**

Find column	Username	Student name	Phone	Email	Student ID	Action
checkbox selection	0934567833	ktuyt@gmail.com	LA0102-190203	<a href="#">Edit</a>	<a href="#">Delete</a>	
Username	Thanh Giang	0934567833	nguyen.giang@gmail.com	LA0102-180204	<a href="#">Edit</a>	<a href="#">Delete</a>

- Query and get detail information if the typo occurred, it can be deleted or updated
- Example: Edit after clicking to Giang student

**Online Logbook.v1**

Edit student Information

Student Name	Thanh Giang Nguyen Tan
Phone Number	+84364002059
Email	nguyen.giang@gmail.com
Address	chánh hòa, chánh an, mang thịt, VN

Submit

- Query and get detailed information if the typo occurred, it can be deleted or updated
- Example: Query teachers

- Add timetable from csv file

	A	B	C	D	E	F	G
1	Week	Day	Time	LA0102-C19006	LA0102-C19004	LA0102-C19005	LA0102-C19001
2	5	1	7:30	NC18NGUVAN10:LA0102-T12111	NC10SINHHOC10:LA0102-T15022	SE19CS311:LA0102-T12001	SHL11HIS101:LA0102-T12111
3	5	1	8:30	NC18NGUVAN10:LA0102-T12111	NC10SINHHOC10:LA0102-T15022	SE19CS311:LA0102-T12001	SHL11HIS101:LA0102-T12111
4	5	1	9:30	NC18HINH11:LA0102-T13220	NC18NGUVAN10:LA0102-T12111	SBE14ECON101:LA0102-T18553	SE19CS311:LA0102-T12001
5	5	1	10:30	NC18HINH11:LA0102-T13220	NC18NGUVAN10:LA0102-T12111	SBE14ECON101:LA0102-T18553	SE19CS311:LA0102-T12001
6	5	1	11:30	NA	NA	NA	NA
7	5	1	13:30	NC10SINHHOC10:LA0102-T15022	NC18HINH11:LA0102-T13220	SHL11HIS101:LA0102-T12111	SBE14ECON101:LA0102-T18553
8	5	1	14:30	NC10SINHHOC10:LA0102-T15022	NC18HINH11:LA0102-T13220	SHL11HIS101:LA0102-T12111	SBE14ECON101:LA0102-T18553
9	5	2	7:30	NA	NA	NA	NA
10	5	2	8:30	NA	NA	NA	NA
11	5	2	9:30	NA	NA	NA	NA
12	5	2	10:30	NA	NA	NA	NA
13	5	2	11:30	NA	NA	NA	NA
14	5	2	13:30	NA	NA	NA	NA
15	5	2	14:30	NA	NA	NA	NA
16	5	3	7:30	NA	NA	NA	NA
17	5	3	8:30	NA	NA	NA	NA
18	5	3	9:30	NA	NA	NA	NA
19	5	3	10:30	NA	NA	NA	NA
20	5	3	11:30	NA	NA	NA	NA
21	5	3	13:30	NA	NA	NA	NA
22	5	3	14:30	NA	NA	NA	NA
23	5	4	7:30	NA	NA	NA	NA
24	5	4	8:30	NA	NA	NA	NA
25	5	4	9:30	NA	NA	NA	NA
26	5	4	10:30	NA	NA	NA	NA
27	5	4	11:30	NA	NA	NA	NA
28	5	4	13:30	NA	NA	NA	NA
29	5	4	14:30	NA	NA	NA	NA

**Online Logbook.v1**

New Timetable

Drop CSV file here or click to upload

TuytGiang Design ©2021

Submit

- Get timetables for the latest version, which can be fixed if there is a typo

**Online Logbook.v1**

Select Options

5 2019 Submit

Date	Time	8A7	9A9	8A1	9A2	
Monday	07:30:00	NC10SINHHOC10	Edit Delete	SHL11HIS101 Edit Delete	NC18NGUVAN10 Edit Delete	SE19CS311 Edit Delete
Monday	08:30:00	NC10SINHHOC10	Edit Delete	SBE14ECON101 Edit Delete	NC18NGUVAN10 Edit Delete	SE19CS311 Edit Delete
Monday	09:30:00	NC18NGUVAN10	Edit Delete	SE19CS311 Edit Delete	NC18HINH11 Edit Delete	SHL11HIS101 Edit Delete
Monday	10:30:00	NC18NGUVAN10	Edit Delete	SE19CS311 Edit Delete	NC18HINH11 Edit Delete	SBE14ECON101 Edit Delete
Monday	11:30:00	NA	Edit Delete	NA Edit Delete	NA Edit Delete	NA Edit Delete
Monday	13:30:00	NC18HINH11	Edit Delete	SBE14ECON101 Edit Delete	NC10SINHHOC10 Edit Delete	SBE14ECON101 Edit Delete
Monday	14:30:00	NC18HINH11	Edit Delete	SBE14ECON101 Edit Delete	NC10SINHHOC10 Edit Delete	SHL11HIS101 Edit Delete

1-8 of 42

- Get timetables for the latest version, which can be fixed if there is a typo

**Online Logbook.v1**

Notifications

Your Information

Management

- Student
- Teacher
- Class
- Timetable
- Create Timetable
- Timetable List

Tutorial I

From week: 5

To Week: -1

Week day: Monday

Time: 07:30:00

Class ID: LA0102-C19004

ClassName: 8A7

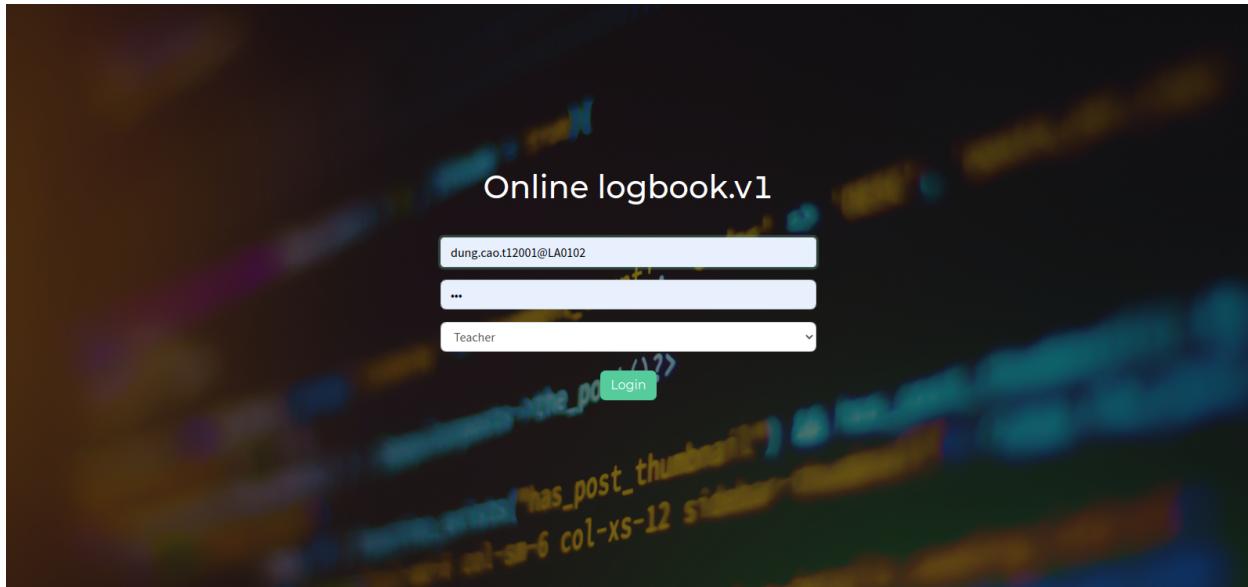
Submit

TuytGiang Design ©2021

## Teacher

### a) Login

- Similar to school account in this function



- Similar to school account in this function

Notifications

Your Information

Management

Tutorial I

The current timetable

Date	Time	8A7	9A9	8A1	9A2
Monday	07:30:00	NC10SINHHOC10	SHL11HIS101	NC18NGUVAN10	SE19CS311
Monday	08:30:00	NC10SINHHOC10	SHL11HIS101	NC18NGUVAN10	SE19CS311
Monday	09:30:00	NC18NGUVAN10	SE19CS311	NC18HINH11	SBE14ECON101
Monday	10:30:00	NC18NGUVAN10	SE19CS311	NC18HINH11	SBE14ECON101
Monday	11:30:00	NA	NA	NA	NA
Monday	13:30:00	NC18HINH11	SBE14ECON101	NC10SINHHOC10	SHL11HIS101
Monday	14:30:00	NC18HINH11	SBE14ECON101	NC10SINHHOC10	SHL11HIS101

1-8 of 42

TuytGiang Design ©2021

## b) Account information

- Similar to school account in this function

Notifications

Your Information

Account Infor

Change Password

Management

Tutorial I

Edit Account Information

Teacher Name	Cao Tiên Dũng
Province	teacher
Major	KHMT
Address	
Phone	987111122
Email	dung.cao@gmail.com

Submit

TuytGiang Design ©2021

- Similar to school account in this function

**Online Logbook.v1**

Update password

Old password  
Type your old password here ..

New password  
Type your new password here ..

Submit

TuytGiang Design ©2021

### c) Management

- Get timetable and teacher's timetable

**Online Logbook.v1**

Select Options

Date	Time	Subject	Teacher	Actions
Monday	07:30:00	NC10SINHHOC10	SHL11HIS101	Edit
Monday	08:30:00	NC10SINHHOC10	SBE14ECON101	Edit
Monday	09:30:00	NC18NGUVAN10	SE19CS311	Edit
Monday	10:30:00	NC18NGUVAN10	SE19CS311	Edit
Monday	11:30:00	NA	NA	Edit
Monday	13:30:00	NC18HINH11	SBE14ECON101	Edit
Monday	14:30:00	NC18HINH11	SBE14ECON101	Edit

1-8 of 42

- Get timetable and teacher's timetable

**Online Logbook.v1**

time	courseCode	day	courseName	classId	className
07:30:00	SE19CS311	Monday	Introduction to Database	LA0102-C19005	9A2
08:30:00	SE19CS311	Wednesday	Introduction to Database	LA0102-C19005	9A2
08:30:00	SE19CS311	Monday	Introduction to Database	LA0102-C19005	9A2
08:30:00	SE19CS311	Monday	Introduction to Database	LA0102-C19005	9A2
09:30:00	SE19CS311	Monday	Introduction to Database	LA0102-C19001	9A9
10:30:00	SE19CS311	Monday	Introduction to Database	LA0102-C19001	9A9

- The teacher's timetable with status value will represent whether the logbook was marked for these lessons. If not click to add

**Online Logbook.v1**

courseName	classId	className ↑	status	Logbook Action
Introduction to Database	LA0102-C19005	9A2	Non active	Edit
Introduction to Database	LA0102-C19005	9A2	Active	Edit
Introduction to Database	LA0102-C19005	9A2	Active	Edit
Introduction to Database	LA0102-C19005	9A2	Non active	Edit
Introduction to Database	LA0102-C19001	9A9	Non active	Edit
Introduction to Database	LA0102-C19001	9A9	Non active	Edit

- The teacher's timetable with status value will represent whether the logbook was marked for these lessons. If not click to add

**Online Logbook.v1**

The screenshot shows a user interface for adding a logbook entry. On the left, a sidebar menu includes: Notifications, Your Information, Account Infor, Change Password, Management (selected), Logbook (selected), Ranking, Timetable (selected), Timetable List, My Timetable (selected), Tutorial !, and a back arrow. The main area has fields for Week (5), Year (2019), Class (9A2), Day (Monday), Course Name (Introduction to Database), Lesson (Introduction to Database), Grade (100), Comment (Good), and Note (Giang is absent again). A green 'Submit' button is at the bottom right. A success message 'sucessfull' is visible in the top right corner.

- Get the logbook of a specific week. Teachers can access different class

**Online Logbook.v1**

The screenshot shows a 'Select Options' dialog with dropdowns for Week (2), Year (2019), Class (9A2), and a 'Submit' button. Below is a table titled 'Week Class Log...' with columns: Date, Time, Course, Lesson, Teacher, and Grade. The table lists Monday entries from 07:30:00 to 14:30:00, all marked as NA for Date, Time, Course, Lesson, Teacher, and Grade. A page navigation bar at the bottom shows '1-8 of 42' with arrows.

Date	Time	Course	Lesson	Teacher	Grade
Monday	07:30:00	NA	NA	NA	NA
Monday	08:30:00	Introduction to Database	Relational Model	NA	100
Monday	09:30:00	NA	NA	NA	NA
Monday	10:30:00	NA	NA	NA	NA
Monday	11:30:00	NA	NA	NA	NA
Monday	13:30:00	NA	NA	NA	NA
Monday	14:30:00	NA	NA	NA	NA

- Get ranking by week and year

**Online Logbook.v1**

The screenshot shows the 'Week Ranking' section of the application. On the left, there's a dark sidebar with navigation links: Notifications, Your Information, Management (with sub-links Logbook, Ranking, Year Ranking, Week Ranking, Timetable), and Tutorial 1. The main content area has a title 'Select Options' with dropdown menus for year (2019), class (9), and a green 'Submit' button. Below this is a table with two rows of data:

<input type="checkbox"/>	className	classId	academicYear	headTeacherName	grade
<input type="checkbox"/>	9A9	LA0102-C19001	2019-2020	Cao Tiến Dũng	0
<input type="checkbox"/>	9A2	LA0102-C19005	2019-2020	Trần Duy Hiển	0

At the bottom right of the content area, it says '1-2 of 2' with navigation arrows.

- Get ranking by week and year

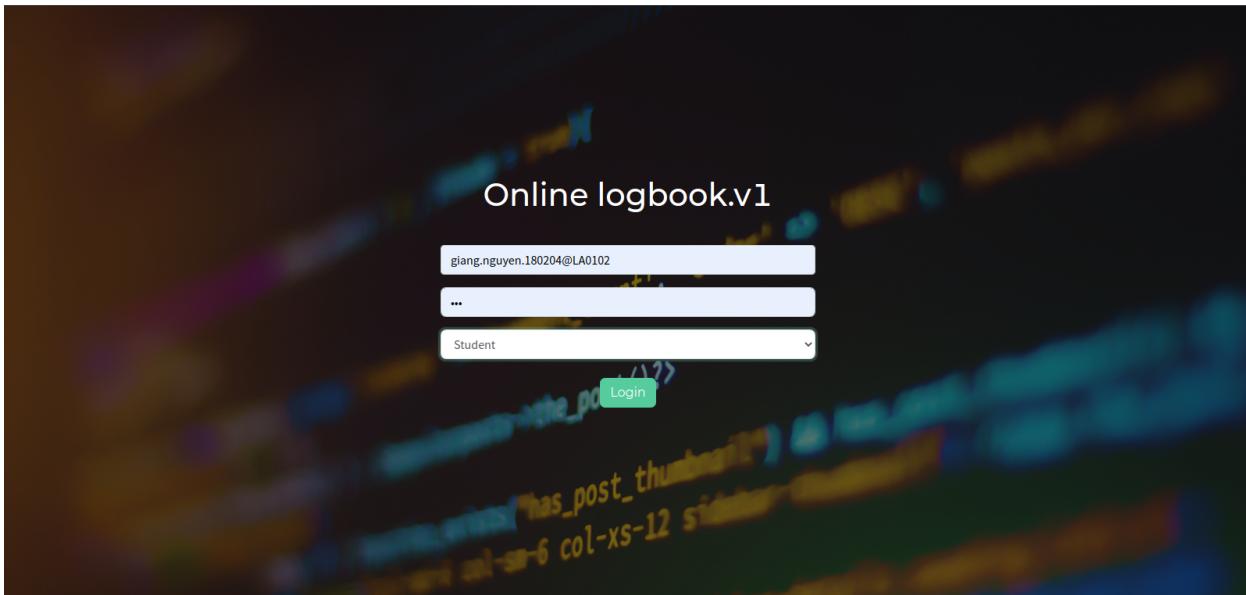
**Online Logbook.v1**

The screenshot shows the 'Year Ranking' section of the application. The sidebar and table structure are identical to the 'Week Ranking' screenshot above, but the dropdown values are different: year 2019, class 9, and a green 'Submit' button. The table data is also the same as the previous screenshot.

## Student

### a) Login:

- Similar to teacher and school



- Similar to teacher and school

**Online Logbook.v1**

A screenshot of the "Online Logbook.v1" dashboard. The top navigation bar includes links for "Notifications", "Your Information", "Management", and "Tutorial". On the right side of the top bar are a user profile icon and a "Logout" link. The main content area is titled "The current timetable" and displays a table of scheduled classes for Monday. The table has columns for Date, Time, and six subject codes (8A7, 9A9, 8A1, 9A2).

Date	Time	8A7	9A9	8A1	9A2
Monday	07:30:00	NC10SINHHOC10	SHL11HIS101	NC18NGUVAN10	SE19CS311
Monday	08:30:00	NC10SINHHOC10	SHL11HIS101	NC18NGUVAN10	SE19CS311
Monday	09:30:00	NC18NGUVAN10	SE19CS311	NC18HINH11	SBE14ECON101
Monday	10:30:00	NC18NGUVAN10	SE19CS311	NC18HINH11	SBE14ECON101
Monday	11:30:00	NA	NA	NA	NA
Monday	13:30:00	NC18HINH11	SBE14ECON101	NC10SINHHOC10	SHL11HIS101
Monday	14:30:00	NC18HINH11	SBE14ECON101	NC10SINHHOC10	SHL11HIS101

TuytGiang Design ©2021

- Similar to teacher and school

**Online Logbook.v1**

User Name: giang.nguyen.180204@LA0102  
Name: Nguyễn Tân Thành Giang  
Email: nguyen.giang@gmail.com  
Student ID: LA0102-180204  
Head Teacher: Trần Duy Hiển  
Class: 9A2  
Address:   
Phone Number: 0934567833

**Submit**

TuytGiang Design ©2021

## b) Account information

- Similar to teacher and school

**Online Logbook.v1**

Old password:   
New password:

**Submit**

TuytGiang Design ©2021

## c) Management

- Similar to teacher and school

## Online Logbook.v1

The screenshot shows a 'Select Options' interface for a timetable. At the top, there are dropdown menus for '2', '2019', and a 'Submit' button. Below is a grid of lesson entries for Monday:

Date	Time	Subject	Teacher	Action
Monday	07:30:00	NC10SINHHOC10	SHL11HIS101	Edit
Monday	08:30:00	NC10SINHHOC10	SHL11HIS101	Edit
Monday	09:30:00	NC18NGUVAN10	SE19CS311	Edit
Monday	10:30:00	NC18NGUVAN10	SE19CS311	Edit
Monday	11:30:00	NA	NA	Edit
Monday	13:30:00	NC18HINH11	SBE14ECON101	Edit
Monday	14:30:00	NC18HINH11	SBE14ECON101	Edit

At the bottom right, it says '1-8 of 42'.

- Similar to teacher and school

## Online Logbook.v1

The screenshot shows a 'Select Options' interface for year ranking. At the top, there are dropdown menus for '2019', '9', and a 'Submit' button. Below is a grid of class details:

className	classId	academicYear	headTeacherName	grade
9A9	LA0102-C19001	2019-2020	Cao Tiến Dũng	300
9A2	LA0102-C19005	2019-2020	Trần Duy Hiển	500

At the bottom right, it says '1-2 of 2'.

## References

- Express/Node introduction - Learn web development | MDN. (2021). Retrieved 27 December 2021, from [https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express\\_Nodejs/Introduction](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction).
- Kumar, R., 2021. *What is Bearer token and How it works?* - DevOpsSchool.com. [online] DevOpsSchool.com. Available at: <<https://www.devopsschool.com/blog/what-is-bearer-token-and-how-it-works/>> [Accessed 26 December 2021].
- Le, H., 2021. *Day hoc truc tuyen de ung pho voi dich COVID-19* | Tạp chí Tuyễn giáo. [online] Tuyengiao.vn. Available at: <<https://tuyengiao.vn/khoa-giao/giao-duc/day-hoc-truc-tuyen-de-ung-pho-voi-dich-covid-19-135538>> [Accessed 26 December 2021].
- Vega, C. (2017). Client-side vs. server-side rendering: why it's not all black and white. Retrieved 27 December 2021, from <https://www.freecodecamp.org/news/what-exactly-is-client-side-rendering-and-hows-it-different-from-server-side-rendering-bd5c786b340d/>
- Vojinovic, I., 2021. *Save Your Data with Empowering Password Statistics* | DataProt. [online] DataProt. Available at: <<https://dataprot.net/statistics/password-statistics/>> [Accessed 26 December 2021].