

**Undergraduate Final Year Project Report**

**PROJECT REPORT**

**Development of eTutoring web-based secure role-based system for Greenwich University**

**Group 3**

**Phan Tuan Khoa (GCS17397),**

**Truong Duc Long (GCS16560),**

**Phan Quang Duc (GCS17019),**

**Ngo Nguyen Song Nguyen (GCS16481),**

**Vo Doan Phuong Duy (GCS16274)**

**Bachelor of Science with Honours in Computing**

**TABLE OF CONTENTS**

[**1.** **Introduction** 3](#_Toc39750926)

[**1.1. Project aim & objectives** 4](#_Toc39750927)

[1.1.1. Analyze the system 4](#_Toc39750928)

[1.1.2. Design the system 4](#_Toc39750929)

[1.1.3. Implementation 5](#_Toc39750930)

[1.1.4. Testing the system 5](#_Toc39750931)

[**1.2. Legal, Social, Ethical and Professional** 5](#_Toc39750932)

[1.2.1. Legal 5](#_Toc39750933)

[1.2.2. Social 6](#_Toc39750934)

[1.2.3. Ethical 6](#_Toc39750935)

[1.2.4. Professional 6](#_Toc39750936)

[**1.3. Planning** 6](#_Toc39750937)

[**2. Database** 8](#_Toc39750938)

[**3. Functionality** 9](#_Toc39750939)

[3.1. Login 9](#_Toc39750940)

[3.2. Admin add new user 10](#_Toc39750941)

[3.3. Admin delete user 11](#_Toc39750942)

[3.4. Admin/Staff add new class 12](#_Toc39750943)

[3.5. Admin/Staff delete class 13](#_Toc39750944)

[3.6. Admin/Staff Add Student, Teacher with Class in Group 14](#_Toc39750945)

[3.7. Admin/Staff delete Group 15](#_Toc39750946)

[3.8. Teacher/Student upload file 16](#_Toc39750947)

[**5. Testing** 17](#_Toc39750948)

[5.1. Test case 17](#_Toc39750949)

[5.2. Test result 20](#_Toc39750950)

[**6. Agile Method** 25](#_Toc39750951)

[6.1. Backlog 25](#_Toc39750952)

[6.2. Meeting minutes 25](#_Toc39750953)

[**2.3.20 Sprint Meeting** 25](#_Toc39750954)

[**16.3.20 Sprint Meeting** 27](#_Toc39750955)

[**30.3.20 Sprint Meeting** 28](#_Toc39750956)

[**13.4.20 Sprint Meeting** 29](#_Toc39750957)

[**27.4.20 Sprint Meeting** 31](#_Toc39750958)

[**7. Source code and screencast** 33](#_Toc39750959)

[7.1. Add Classrooms 33](#_Toc39750960)

[7.2. Groups 36](#_Toc39750961)

[7.3. Dashboard 38](#_Toc39750962)

[7.4. Function Chat Groups 40](#_Toc39750963)

[7.5. Upload Files 45](#_Toc39750964)

[**Appendix** 50](#_Toc39750965)

[**TABLE OF FIGURES** 51](#_Toc39750966)

[**TABLE OF REFERENCES** 51](#_Toc39750967)

**INTRODUCTION: HOW TO OPERATE THE WEBSITE**

Before you running project Laravel, please running commander from the Laravel putting this into **artisan command: “PHP artisan storage: link”**

1. **Memorize:**

**Admin:** Will maintain to authorize access and create new for Staff, Teacher, Student and each of them would have a difference ID to access the website under managing by Admin

* **Note:**Admin is the only one authorizer can access the website and he could use the (create, update, delete) also he could update new (ID, password) for (Staff, teacher, student)

**Staff:**His jobs only add a member to the group of that class

**Teacher:** His only has the chat function and upload function

**Student:** This guy only has the chat function vs upload function

* Note: The dashboard is only showing up the group box only for teacher and student

1. **Manufacture**

**Class:** Class must be creating by the Teacher, after that, the teacher could add any student into that class

**Chat Function:** Only student and teacher can see all information and files by their ID from that group by that teacher upload

**Upload:** Choosing the right class and right file to upload into the group and only the Teacher can upload new files to the right files from his authorize.

* Note: Students only can see his files and teacher files from that group he learns. About the teacher, he can see all files upload by himself and his student’s files

**Attention:** Do not let’s file empty because it only upload PDF files with the full name of it.

# **Introduction**

Greenwich University wants to develop a web-based, role-based eTutoring system to aid students in learning more effectively. After some discussions internally, they want to implement Scrum with Agile concepts in this project and choose us to be the scrum team that will handle this project.

We are a small group of students tasked with building the system for Greenwich University. In our scrum team, we have:

**Long** – the ScrumMaster. He works with documents and finalize the report.

**Khoa** – works together with the ScrumMaster and other team members to handle documents.

**Duc** – the database designer. He is in charge of designing the database and connect it with the front-end.

**Duy** – the programmer and tester who works together with Duc to handle the back-end. He has some professional experience in working as a professional software developer.

**Nguyen** – the web designer who handles the front-end. He has some professional experience in working as a professional software developer.

In this project, we will develop the system using PHP and MySQL for the back-end. Both of them are free and open-source technologies that will significantly reduce the development cost, comparing to Microsoft SQL with its expensive pricing. Combining with XAMPP which acts as a local host or a local server, we can quickly test the system and reduce the development cost further. For the front-end, we use the common HTML, CSS and JavaScript. All of which are popular technologies with tons of supports from their communities. And we decided to use Lavarel for its easy database migration, simple and clean code that makes it easy to maintain, and being a free open-source framework. We especially love the integrated security measures that helps prevent attacks such as SQL injection and CSR forgery.

In this system, there will be three types of users: Students, Tutors, and Staffs. **The Students** can use the system for meetings with their tutors, upload their documents and commenting on them, blogging, and have their own personal dashboard that summarize their interactions.

**The Tutors** will also be able to use the system for meetings with their students, uploading documents and commenting on them and blogging. The interesting thing is that they will have a dashboard of their personal students. It can be sorted and filtered the way they want.

**The Staffs** have the highest authorities. They can allocate and reallocate personal tutors to students and both the students and tutors will receive notification emails when this happens. The allocation can be done in bulk of ten at a time. Certain authorized staffs can gain access to dashboards of other staff as well as individual dashboards for students.

# **1.1. Project aim & objectives**

With this project, we want to learn how to develop an eTutoring system as a team using Scrum with Agile concepts in 6 months.

In order to achieve our aim, we will define the objectives and describe them in details below.

## 1.1.1. Analyze the system

We start with defining the requirements and functions of the system based on the specification given to us. By defining them, we understand how the system works and decide on the appropriate approach for this project. The requirements of the system as well as technologies and methodologies chosen by us will affect our plans for the project. This shouldn’t take more than two weeks.

The deliverables for this objective would be the backlog, where all of the requirements and tasks to develop the finished product are listed and prioritized.

## 1.1.2. Design the system

After analyzing the system and planning out the project, we start designing the system. By designing the system, we can code according to our design. This can take from two to four weeks.

The deliverables for this objective would be wireframes for the front-end, ERDs and Activity Diagrams for back-end.

## 1.1.3. Implementation

After making the design, we will start coding accordingly to create a functional system with working functions. The system will be finished before the deadline arrives.

The deliverables for this objective would be the source code and screencasts to show the system in action.

## 1.1.4. Testing the system

It is important to test the system that we develop to ensure that they are working as intended. Within every sprint, we test the functions that we are developing.

The deliverables for this objective would be a test plan, a test description, and a test report.

# **1.2. Legal, Social, Ethical and Professional**

## 1.2.1. Legal

PHP is a free open-source scripting language under the PHP License. The project is in accordance to the mentioned license.

MySQL is an open-source relational database management system. The project is in accordance to the MySQL Commercial License.

XAMPP is a compilation of free software that is free of charge and free to copy under the terms of the GNU General Public License. The project is in accordance to the mentioned license.

The Lavarel framework is an open-sources software licensed under the MIT License. The project is in accordance to the mentioned license.

JavaScript is a programming language licensed under the GNU General Public License. The project is in accordance to the mentioned license.

HTML 5 is a markup language licensed under the W3C License. The project is in accordance to the mentioned license.

This project belongs to our team and University of Greenwich. In this project, every materials that are used here will come from free sources or legal copies.

## 1.2.2. Social

The system will be used for the benefits of the University of Greenwich, the tutors and students.

Measures will be taken to ensure the security and privacy of the parties involved.

## 1.2.3. Ethical

This project is the work of our team and other involved parties. There will be no plagiarism in this project.

We ensure the privacy of the parties involved and prevent unauthorized access as well as inappropriate use of data.

We will put in our best effort to prevent cybercrimes that can be committed using our system.

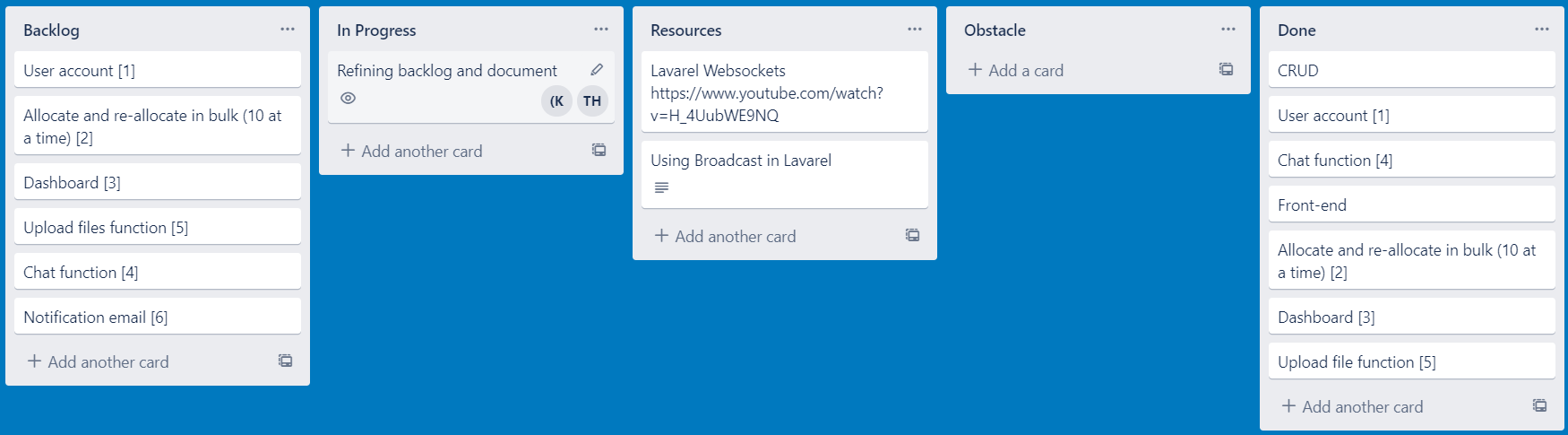
## 1.2.4. Professional

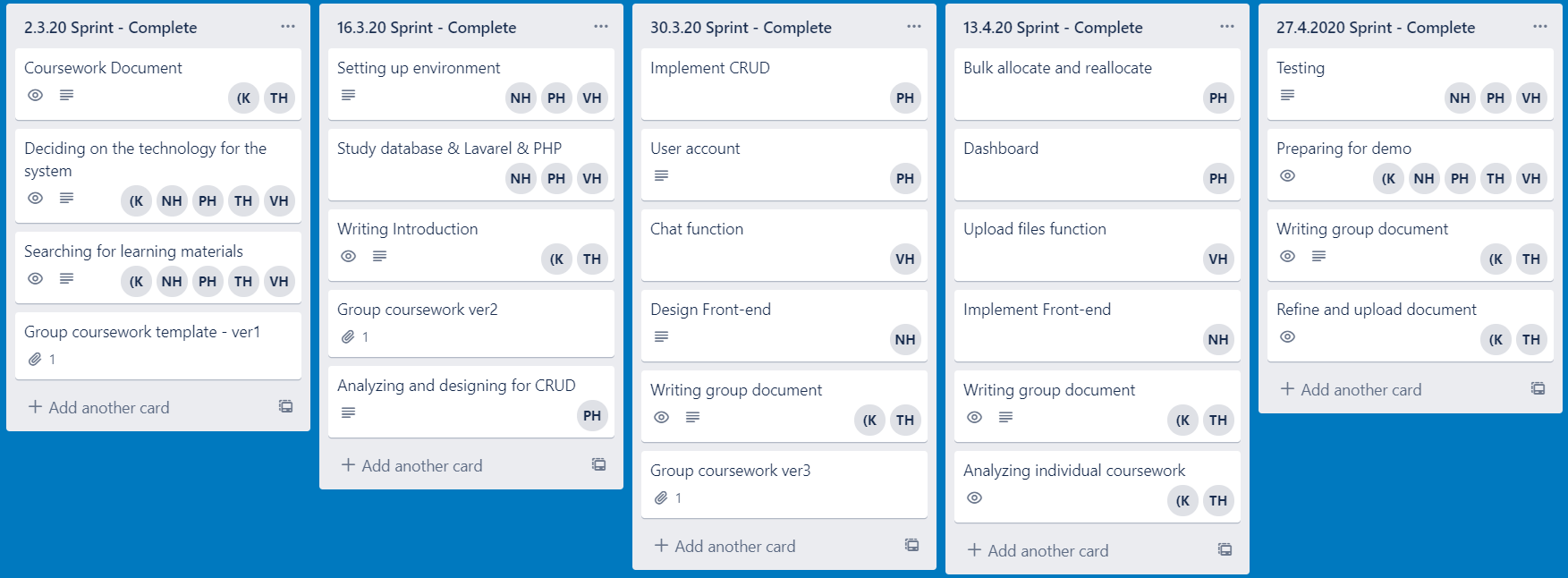
Confidential information will not be disclosed or use for personal gain or a third party’s benefit without permission from the University of Greenwich.

We will seek and accept honest criticism about our work, as well as respect and value alternative perspectives to continuously improve ourselves.

We ensure that the code that we write is tested thoroughly and meets our quality standards. We will also continuously improve the code to keep up the quality.

# **1.3. Planning**





Our plan for this project is to analyze and understand the specification for the system for the first sprint. In the second sprint, we start to setup the environment and design the database. In the third and fourth sprint, we implement all of the main functions of the system and conduct small tests to ensure the quality. Finally, we wrap up the project with a large and thorough test to find and fix errors to ensure a working prototype for demonstration.

# **2. Database**



Figure 1 ERD

# **3. Functionality**

## 3.1. Login



## 3.2. Admin add new user



## 3.3. Admin delete user



## 3.4. Admin/Staff add new class



## 3.5. Admin/Staff delete class



## 3.6. Admin/Staff Add Student, Teacher with Class in Group



## 3.7. Admin/Staff delete Group



## 3.8. Teacher/Student upload file



# **5. Testing**

## 5.1. Test case

|  |  |  |  |
| --- | --- | --- | --- |
| Case | Description | Expected Outcome | Actual Outcome |
| Login | User input Valid Email and Password | Login successfully and move to dashbroad. | Login successfully and move to dashbroad. |
| User input invalid Email and Password | Login fail and back to login form with error message. | Login fail and back to login form with error message. |
| Staff input Valid Email and Password | Login successfully and move to Staff dashbroad. | Login successfully and move to Staff dashbroad. |
| Teacher input Valid Email and Password | Login successfully and move to Teacher dashbroad. | Login successfully and move to Teacher dashbroad. |
| Student input Valid Email and Password | Login successfully and move to Student dashbroad. | Login successfully and move to Student dashbroad. |
| Menu | After Login, the menu will show based on the role that admin set for the user | - Show everything for role Admin  - Show dashboard, Classroom, Group, Chat for Staff  - Show dashboard, Classroom, Group, Chat, Upload File for Teacher  - Show dashboard, Chat, Upload File for Student | - Show everything for role Admin  - Show dashboard, Classroom, Group, Chat for Staff  - Show dashboard, Classroom, Group, Chat, Upload File for Teacher  - Show dashboard, Chat, Upload File for Student |
| View list user | Admin click into the User management section | Show list of user | Show list of user |
| Add new user | Admin input Valid Email and Password | Add new successfully and move to list user. | Add new successfully and move to list user. |
| User input invalid Email and Password | Add new fail and back to login form with error message. | Add new fail and back to login form with error message. |
| Edit User | Admin edit new role for user | Edit success and return to list user with the updated user role | Edit success and return to list user with the updated user role |
| Delete user | Admin choose user and click Delete | Appear popup to confirm user, user click Yes Delete then delete the user return to list without the deleted user | Appear popup to confirm user, user click Yes Delete the return to list without the deleted user |
| Admin choose user and click Delete | Appear popup to confirm user, user click Close then return to list with full of user | Appear popup to confirm user, user click Close then return to list with full of user |
| View list Classroom | User click into the Classroom section | Show list of Classroom | Show list of Classroom |
| Add new Classroom | User input Classroom name | Add new successfully and move to list Classroom. | Add new successfully and move to list Classroom. |
| Delete Classroom | User choose Classroom and click Delete | Appear popup to confirm user, user click Yes then delete the classroom and return to list without the deleted classroom | Appear popup to confirm user, user click Yes then delete the classroom and return to list without the deleted classroom |
| User choose Classroom and click Delete | Appear popup to confirm user, user click Close then return to list with full of classroom | Appear popup to confirm user, user click Close then return to list with full of classroom |
| View list Group | User click into the Group section | Show list of Classroom | Show list of Classroom |
| Add new Group | User input Group’s Class, Student and Teacher | Add new successfully and move to list Group. | Add new successfully and move to list Group. |
| Delete Group | User choose Group and click Delete | Appear popup to confirm user, user click Yes then delete the Group and return to list without the deleted Group | Appear popup to confirm user, user click Yes then delete the Group and return to list without the deleted Group |
| User choose Group and click Delete | Appear popup to confirm user, user click Close then return to list with full of Group | Appear popup to confirm user, user click Close then return to list with full of Group |
| Chat | Teacher and Student click to chat section | Show the group of class they participate in and the number of unread message the teacher and student that class have | Show the group of class they participate in and the number of unread message the teacher and student that class have |
| Upload File | Teacher and Student click to chat Upload File section | Show the class they participate in and the teacher uploaded file and their uploaded file but not the other student uploaded file | Show the class they participate in and the teacher uploaded file and their uploaded file but not the other student uploaded file |
|  | Teacher and Student choose the file to upload | Done choosing file and click upload, the uploaded file appear to the list | Done choosing file and click upload, the uploaded file appear to the list |

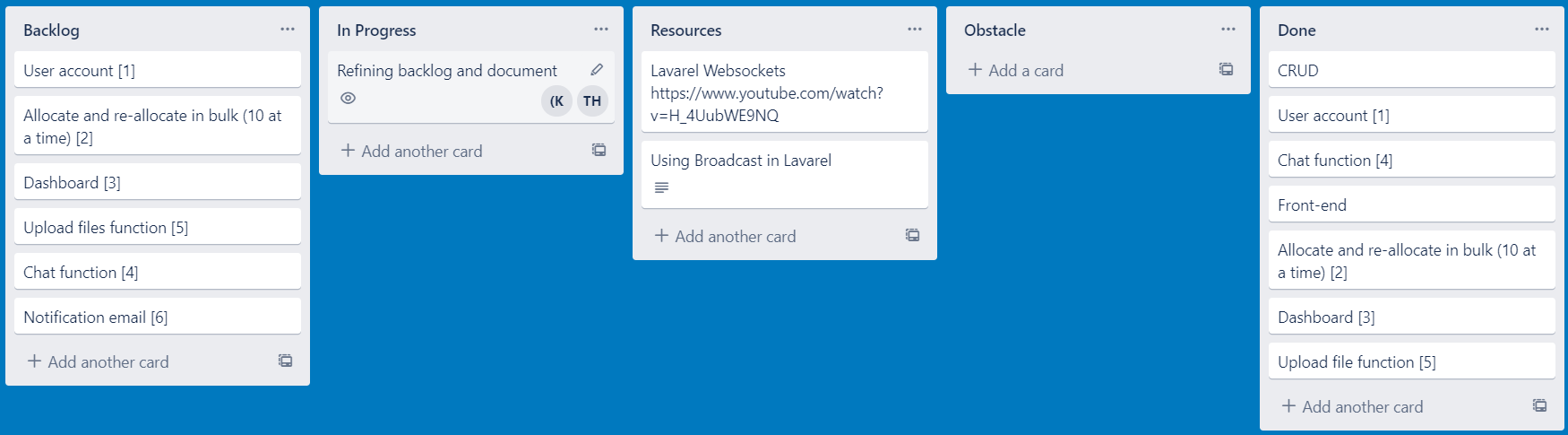
## 5.2. Test result

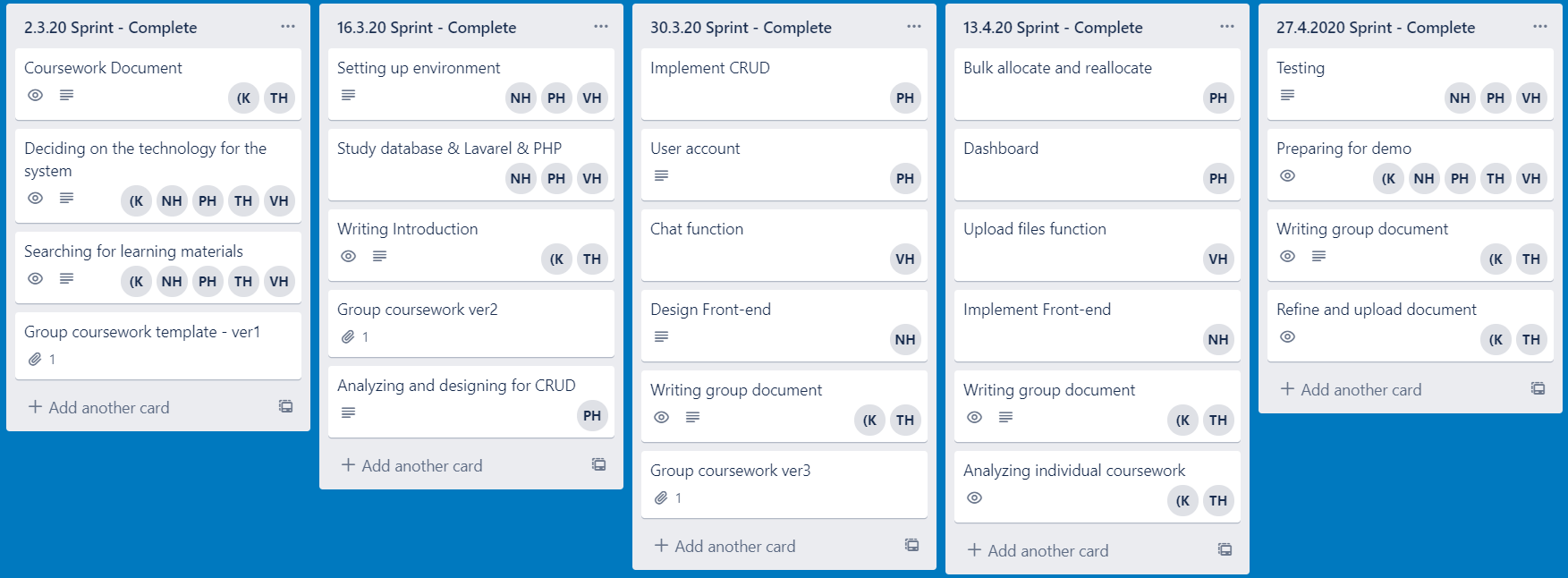
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Case | Description | Expected Outcome | Actual Outcome | Result |
| Login | User input Valid Email and Password | Login successfully and move to dashbroad. | Login successfully and move to dashbroad. | **Pass** |
| User input invalid Email and Password | Login fail and back to login form with error message. | Login fail and back to login form with error message. | **Pass** |
| Staff input Valid Email and Password | Login successfully and move to Staff dashbroad. | Login successfully and move to Staff dashbroad. | **Pass** |
| Teacher input Valid Email and Password | Login successfully and move to Teacher dashbroad. | Login successfully and move to Teacher dashbroad. | **Pass** |
| Student input Valid Email and Password | Login successfully and move to Student dashbroad. | Login successfully and move to Student dashbroad. | **Pass** |
| Menu | After Login, the menu will show based on the role that admin set for the user | - Show everything for role Admin  - Show dashboard, Classroom, Group, Chat for Staff  - Show dashboard, Classroom, Group, Chat, Upload File for Teacher  - Show dashboard, Chat, Upload File for Student | - Show everything for role Admin  - Show dashboard, Classroom, Group, Chat for Staff  - Show dashboard, Classroom, Group, Chat, Upload File for Teacher  - Show dashboard, Chat, Upload File for Student | **Pass** |
| View list user | Admin click into the User management section | Show list of user | Show list of user | **Pass** |
| Add new user | Admin input Valid Email and Password | Add new successfully and move to list user. | Add new successfully and move to list user. | **Pass** |
| User input invalid Email and Password | Add new fail and back to login form with error message. | Add new fail and back to login form with error message. | **Pass** |
| Edit User | Admin edit new role for user | Edit success and return to list user with the updated user role | Edit success and return to list user with the updated user role | **Pass** |
| Delete user | Admin choose user and click Delete | Appear popup to confirm user, user click Yes Delete then delete the user return to list without the deleted user | Appear popup to confirm user, user click Yes Delete the return to list without the deleted user | **Pass** |
| Admin choose user and click Delete | Appear popup to confirm user, user click Close then return to list with full of user | Appear popup to confirm user, user click Close then return to list with full of user | **Pass** |
| View list Classroom | User click into the Classroom section | Show list of Classroom | Show list of Classroom | **Pass** |
| Add new Classroom | User input Classroom name | Add new successfully and move to list Classroom. | Add new successfully and move to list Classroom. | **Pass** |
| Delete Classroom | User choose Classroom and click Delete | Appear popup to confirm user, user click Yes then delete the classroom and return to list without the deleted classroom | Appear popup to confirm user, user click Yes then delete the classroom and return to list without the deleted classroom | **Pass** |
| User choose Classroom and click Delete | Appear popup to confirm user, user click Close then return to list with full of classroom | Appear popup to confirm user, user click Close then return to list with full of classroom | **Pass** |
| View list Group | User click into the Group section | Show list of Classroom | Show list of Classroom | **Pass** |
| Add new Group | User input Group’s Class, Student and Teacher | Add new successfully and move to list Group. | Add new successfully and move to list Group. | **Pass** |
| Delete Group | User choose Group and click Delete | Appear popup to confirm user, user click Yes then delete the Group and return to list without the deleted Group | Appear popup to confirm user, user click Yes then delete the Group and return to list without the deleted Group | **Pass** |
| User choose Group and click Delete | Appear popup to confirm user, user click Close then return to list with full of Group | Appear popup to confirm user, user click Close then return to list with full of Group | **Pass** |
| Chat | Teacher and Student click to chat section | Show the group of class they participate in and the number of unread message the teacher and student that class have | Show the group of class they participate in and the number of unread message the teacher and student that class have | **Pass** |
| Upload File | Teacher and Student click to chat Upload File section | Show the class they participate in and the teacher uploaded file and their uploaded file but not the other student uploaded file | Show the class they participate in and the teacher uploaded file and their uploaded file but not the other student uploaded file | **Pass** |
|  | Teacher and Student choose the file to upload | Done choosing file and click upload, the uploaded file appear to the list | Done choosing file and click upload, the uploaded file appear to the list | **Pass** |

# **6. Agile Method**

## 6.1. Backlog

Trello link: <https://trello.com/b/uo2jGsDL>





## 6.2. Meeting minutes

Due to the event of COVID-19, we could not carry out some of the Agile and Scrum concepts in meetings. The meetings were conducted online and had some limitations comparing to meeting in person.

### **2.3.20 Sprint Meeting**

**Information / Decisions**

|  |  |
| --- | --- |
| **No.** | **Item** |
| **01** | The product owner has requested the team to develop the system with the specifications. |
| **02** | The team has agreed to look into the specifications and analyze it. |
| **03** | The product owner has requested the team to decide on the technologies used for this project. |
| **04** | The team has agreed to recommend the appropriate technology for the project based on the requirements and team’s capabilities. |
| **05** | The team has agreed to search for learning materials. |
| **06** | Khoa and Long have agreed to work on documentation |

**Action Items**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Item** | **Who** | **By** |
| **1.1** | Discussing project requirements | Team, Product Owner |  |
| **1.2** | Deciding on the technology for the system | Team, Product Owner | 13/03/2020 |
| **1.3** | Searching for learning materials | Team | 13/03/2020 |
| **1.4** | Writing documentation | Khoa, Long | 13/03/2020 |
| **Note:** |  |  |  |
|  | Documented by: Phan Tuan Khoa |  |  |

### **16.3.20 Sprint Meeting**

**Information / Decisions**

|  |  |
| --- | --- |
| **No.** | **Item** |
| **01** | The product owner has requested the team to develop the system. |
| **02** | The team has agreed to do CRUD. |
| **03** | Nguyen, Duc and Duy have agreed to set up the environment. |
| **04** | Nguyen, Duc and Duy have agreed to study database, Lavarel and PHP. |
| **05** | Khoa and Long have agreed to continue write documentation. |
| **06** | Duc has agreed to analyze and design for CRUD. |

**Action Items**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Item** | **Who** | **By** |
| **1.1** | Discussing project requirements | Team, Product Owner |  |
| **1.2** | Setting up environment | Nguyen, Duc, Duy | 27/03/2020 |
| **1.3** | Studying database, Lavarel and PHP | Nguyen, Duc, Duy | 27/03/2020 |
| **1.4** | Write documentation | Khoa, Long | 27/03/2020 |
| **1.5** | Analyzing and designing for CRUD | Duc | 27/03/2020 |
| **Note:** |  |  |  |
|  | Documented by: Phan Tuan Khoa |  |  |

### **30.3.20 Sprint Meeting**

**Information / Decisions**

|  |  |
| --- | --- |
| **No.** | **Item** |
| **01** | The product owner has requested the team to implement CRUD, user account, chat function, and front-end. |
| **02** | The team has agreed to implement CRUD, user account, chat function, and design the front-end. |
| **03** | Duc has agreed to implement CRUD. |
| **04** | Duc has agreed to implement user account. |
| **05** | Duy has agreed to implement chat function. |
| **06** | Nguyen has agreed to design the front-end. |
| **07** | Khoa and Long have agreed to write document. |

**Action Items**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Item** | **Who** | **By** |
| **1.1** | Discussing project requirements | Team, Product Owner |  |
| **1.2** | Implement CRUD | Duc | 10/04/2020 |
| **1.3** | User account | Duc | 10/04/2020 |
| **1.4** | Chat function | Duy | 10/04/2020 |
| **1.5** | Design Front-end | Nguyen | 10/04/2020 |
| **1.6** | Writing group document | Khoa, Long | 10/04/2020 |
| **Note:** |  |  |  |
|  | Documented by: Phan Tuan Khoa |  |  |

### **13.4.20 Sprint Meeting**

**Information / Decisions**

|  |  |
| --- | --- |
| **No.** | **Item** |
| **01** | The product owner has requested the team to implement bulk allocation and reallocation, dashboard, upload files function and the front-end. |
| **02** | The team has agreed to implement bulk allocation and reallocation, dashboard, upload files function and the front-end. |
| **03** | Duc has agreed to implement bulk allocation and reallocation. |
| **04** | Duc has agreed to implement the dashboard. |
| **05** | Duy has agreed to implement the upload files function. |
| **06** | Nguyen has agreed to implement the front-end. |
| **07** | Khoa and Long has agreed to write documentation and analyze individual coursework. |

**Action Items**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Item** | **Who** | **By** |
| **1.1** | Discussing project requirements | Team, Product Owner |  |
| **1.2** | Bulk allocation and reallocation | Duc | 24/04/2020 |
| **1.3** | Dashboard | Duc | 24/04/2020 |
| **1.4** | Upload files function | Duy | 24/04/2020 |
| **1.5** | Implement the front-end | Nguyen | 24/04/2020 |
| **1.6** | Writing documentation | Khoa, Long | 24/04/2020 |
| **1.7** | Analyzing individual coursework | Khoa, Long | 24/04/2020 |
| **Note:** |  |  |  |
|  | Documented by: Phan Tuan Khoa |  |  |

### **27.4.20 Sprint Meeting**

**Information / Decisions**

|  |  |
| --- | --- |
| **No.** | **Item** |
| **01** | The product owner has requested the team to test the system thoroughly, prepare for demo, and upload the documentation. |
| **02** | The team has agreed to test the system, prepare for the demo, and upload the documentation. |
| **03** | Nguyen, Duc and Duy have agreed to test the system. |
| **04** | The team has agreed to prepare for the demo. |
| **05** | Khoa and Long has agreed to write the document. |
| **06** | Khoa and Long has agreed to refine and upload the document. |

**Action Items**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Item** | **Who** | **By** |
| **1.1** | Discussing project requirements | Team, Product Owner |  |
| **1.2** | Testing | Nguyen, Duc, Duy | 07/05/2020 |
| **1.3** | Preparing for demo | Team | 07/05/2020 |
| **1.4** | Writing group document | Khoa, Long | 07/05/2020 |
| **1.5** | Refine and upload document | Khoa, Long | 07/05/2020 |
| **Note:** |  |  |  |
|  | Documented by: Phan Tuan Khoa |  |  |

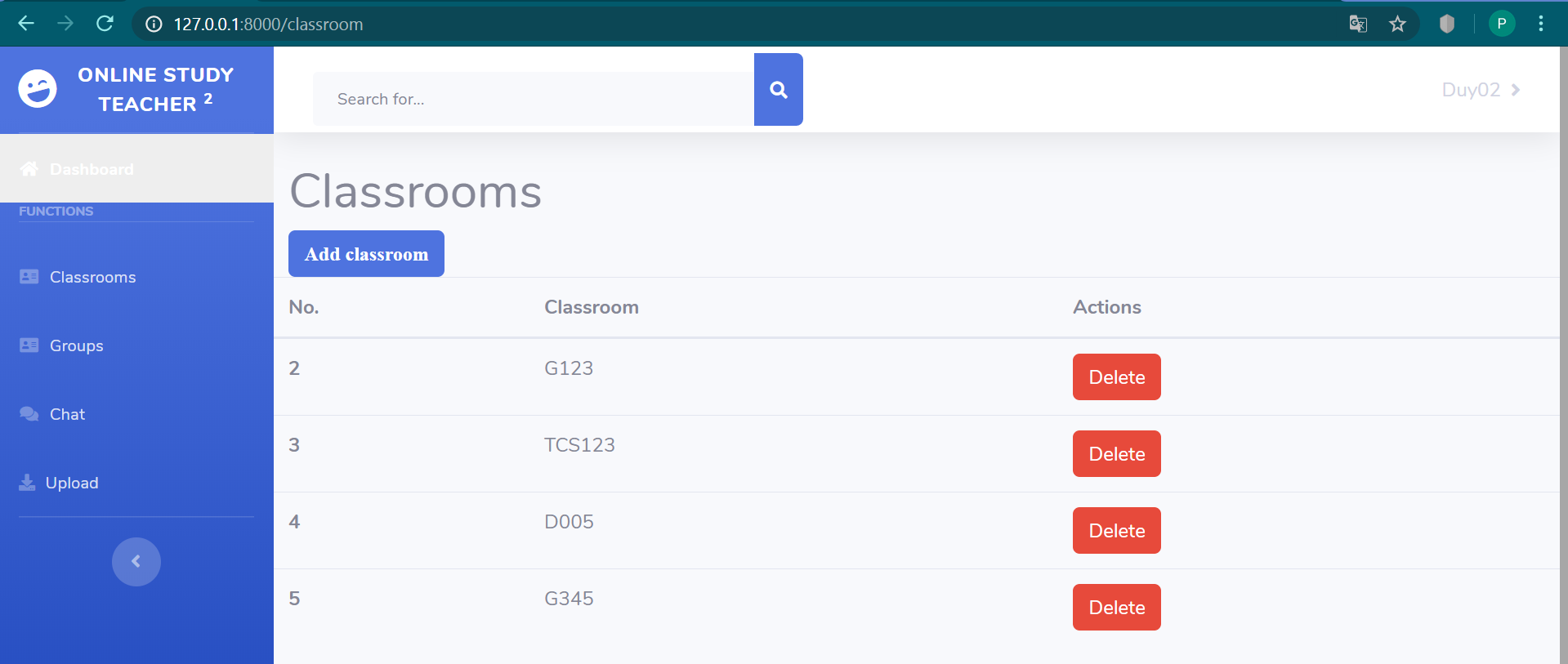
Link GitHub: <https://github.com/ashblake30194/COMP1640>

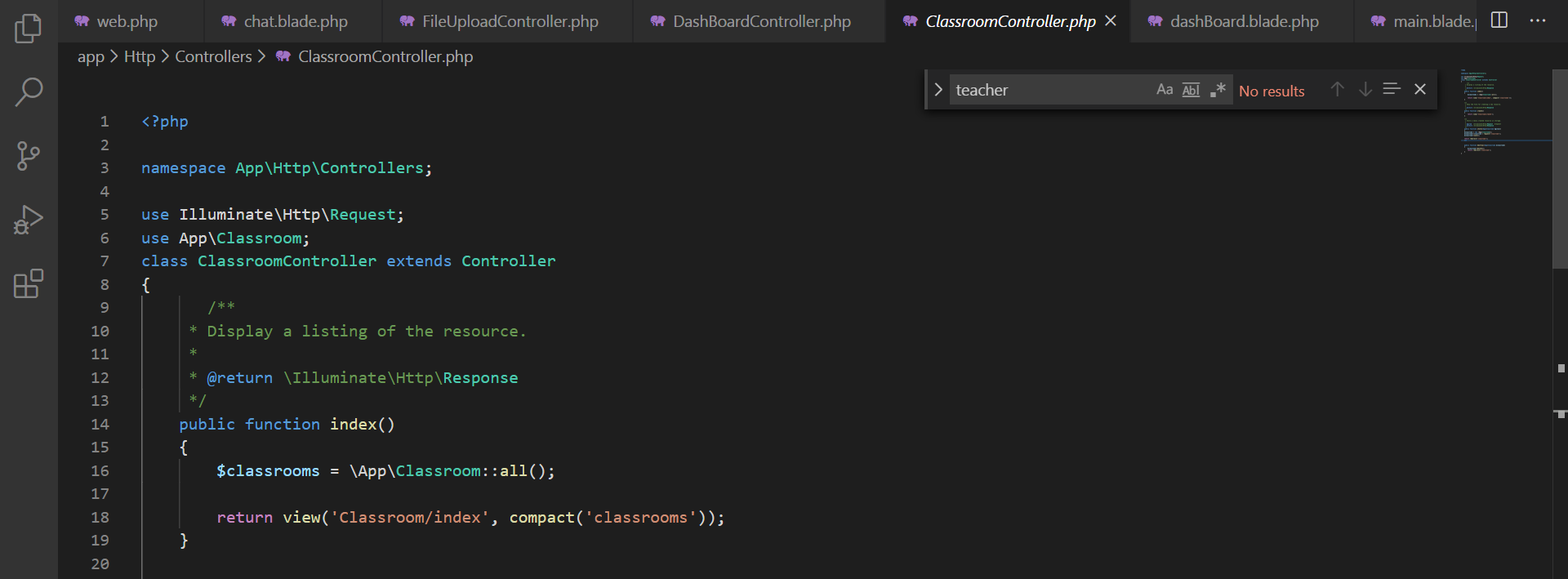
Daily Scrum:

# **7. Source code and screencast**

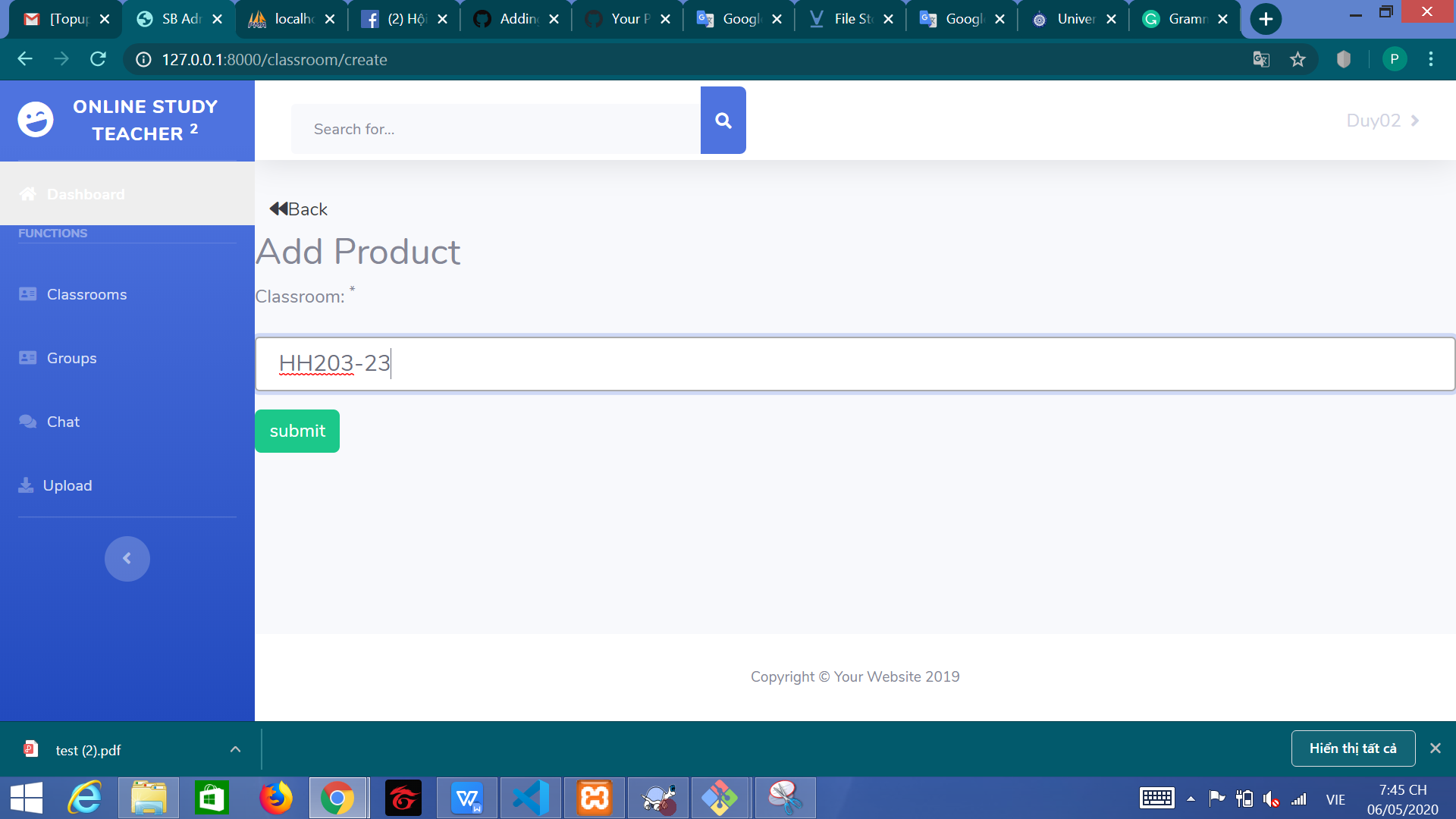
Link for source code, video and presentation: <https://drive.google.com/drive/folders/1htBXznbfigIRqXyZmwxtlmz1H3W1apVu?usp=sharing>

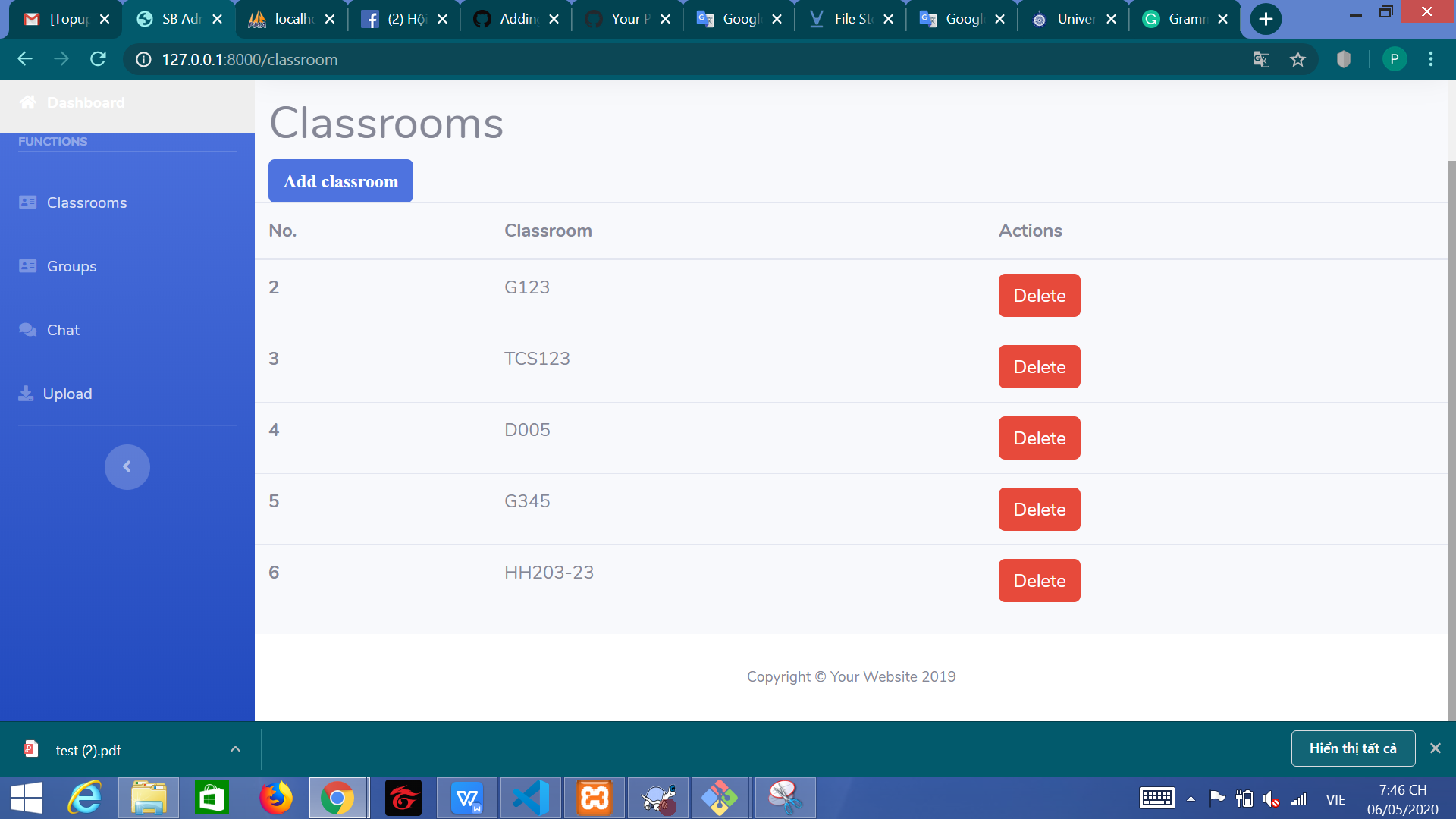
## 7.1. Add Classrooms

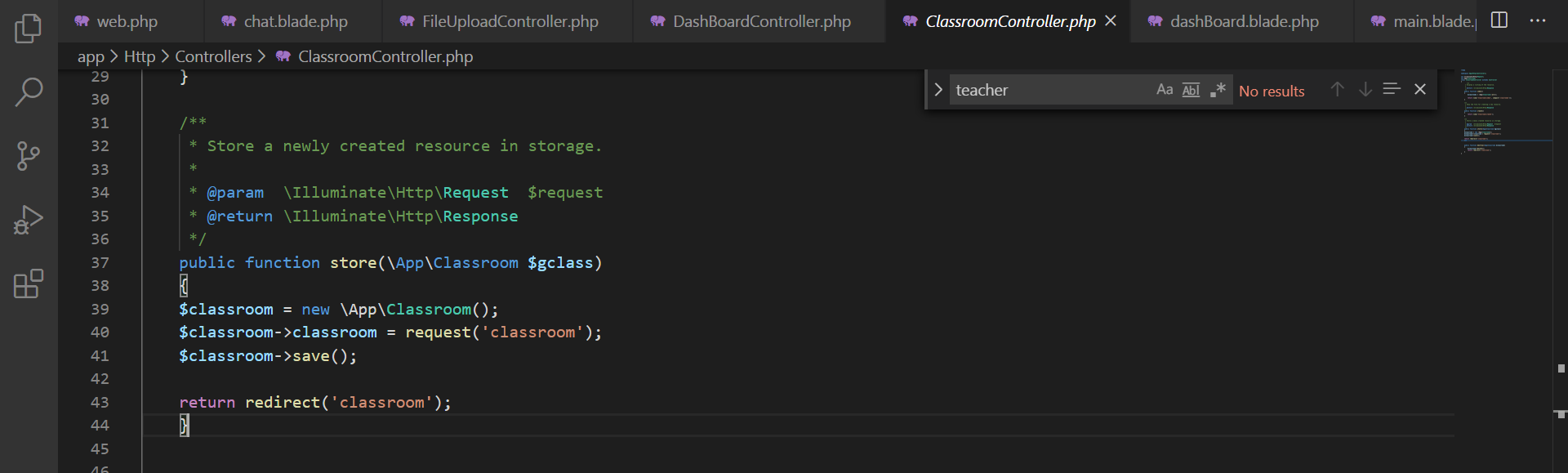




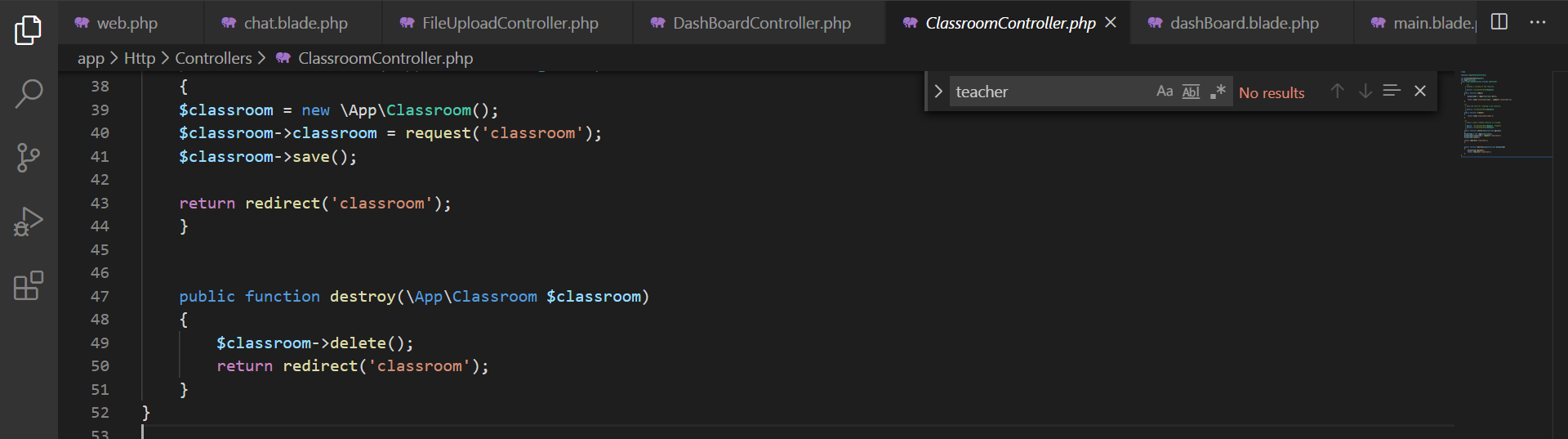
This function is for getting data in the database and showing lists of classrooms.

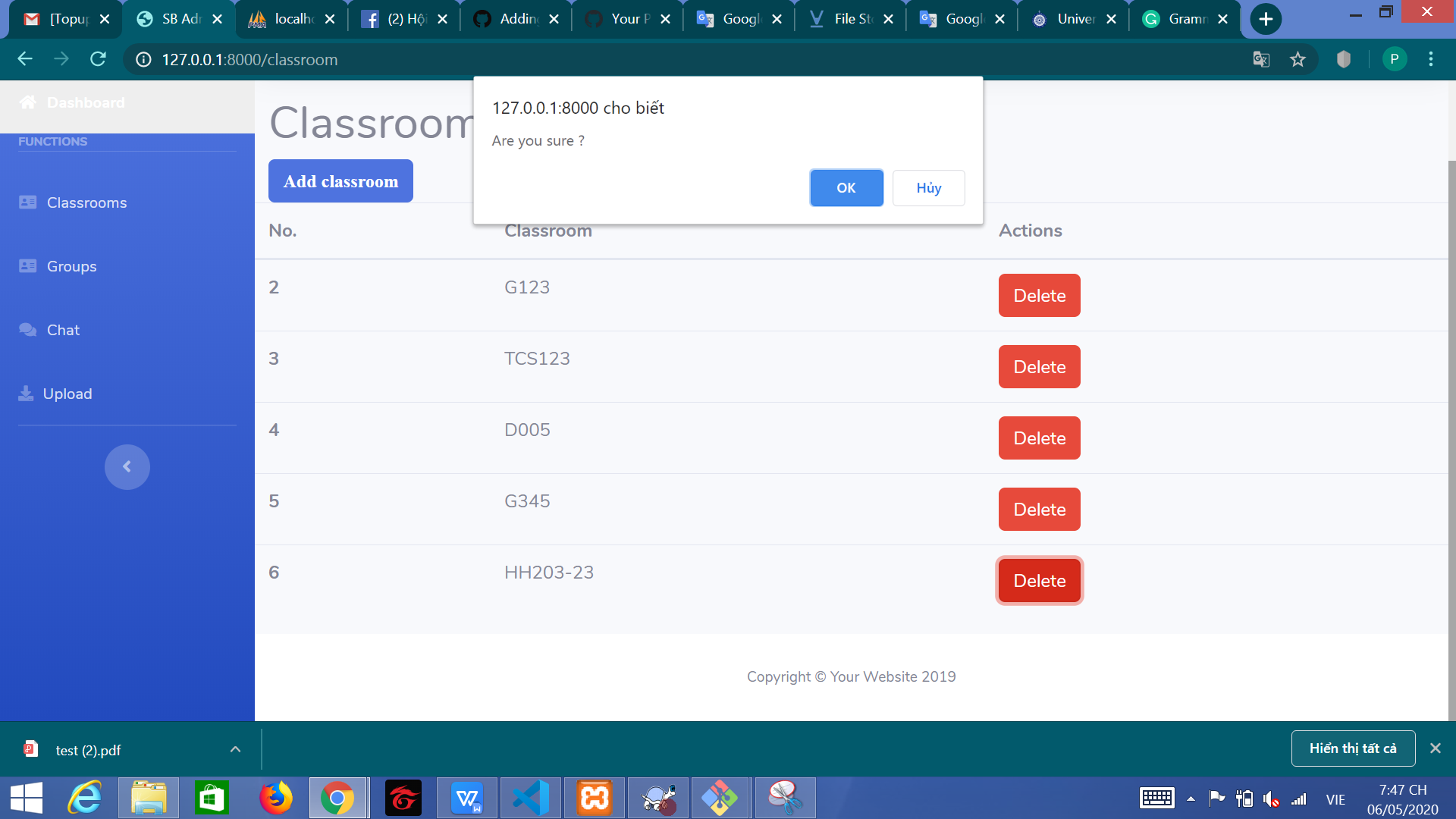


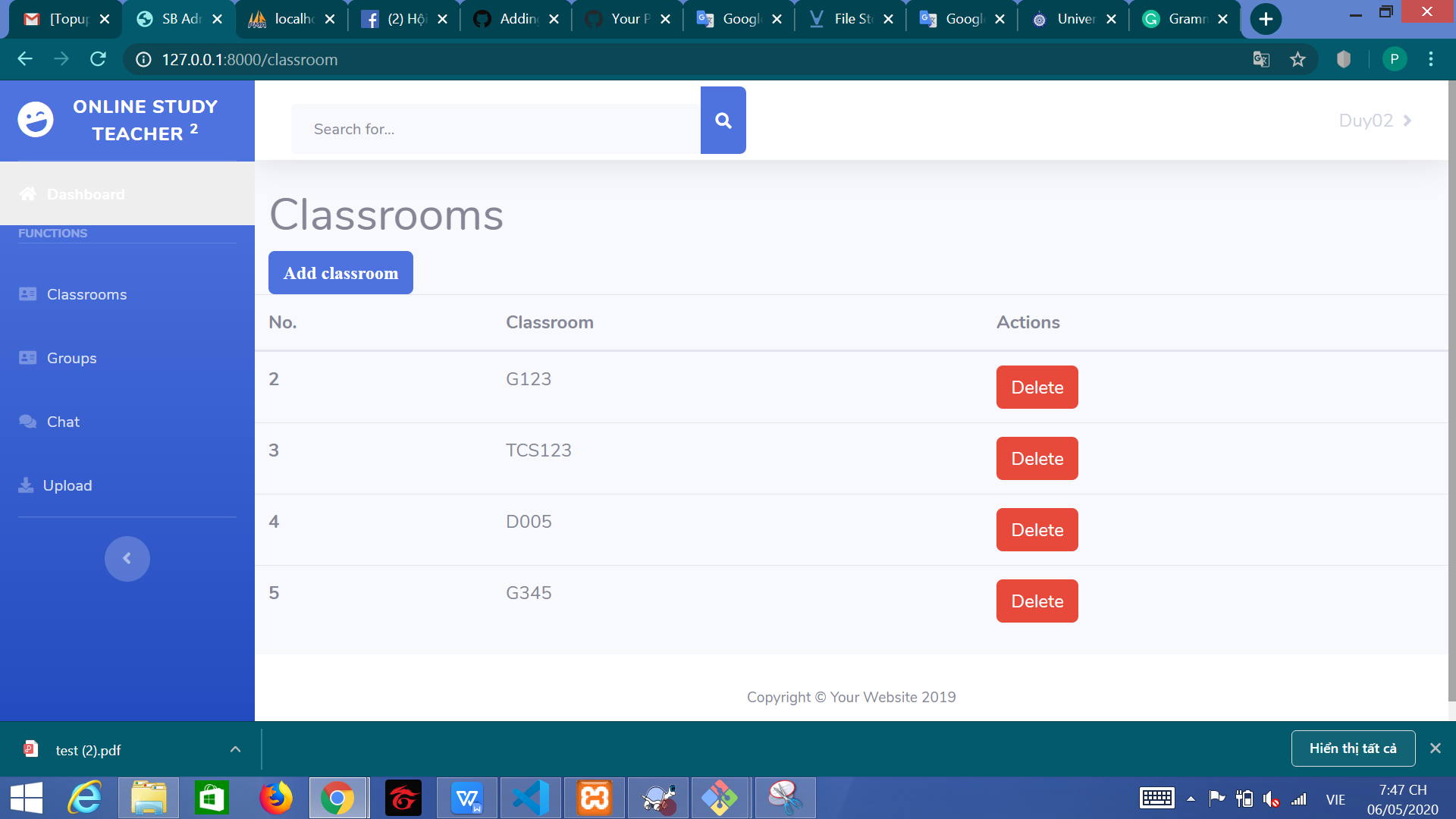




This function allows you to create new classrooms.

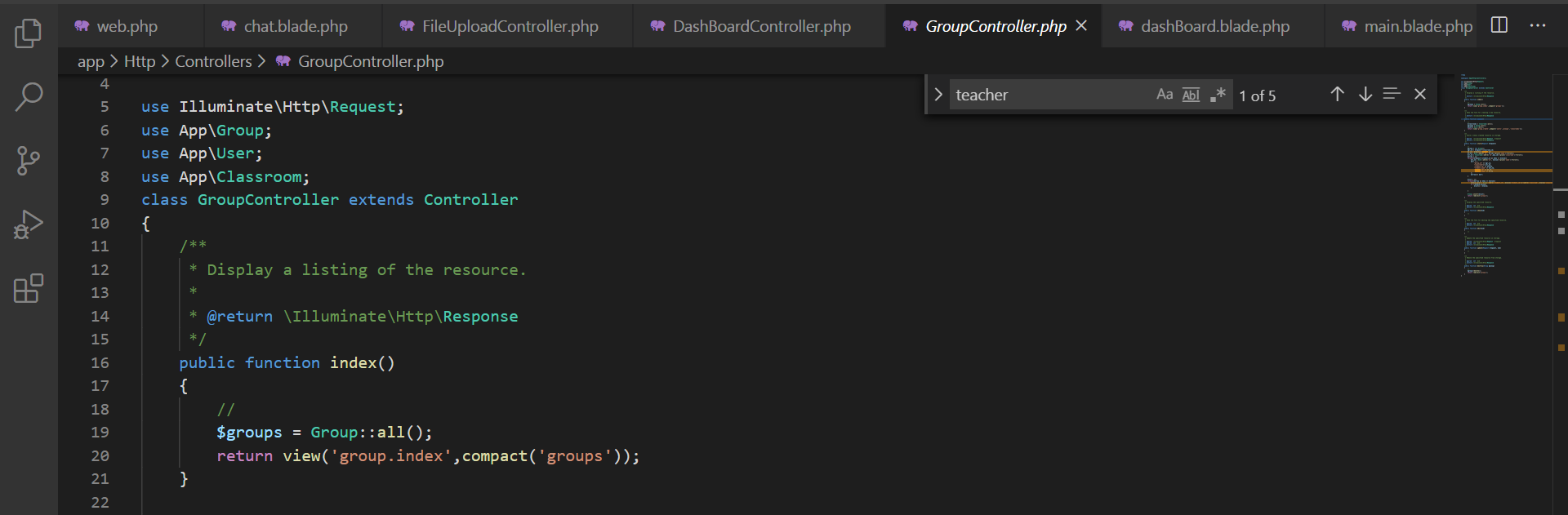






You can delete classrooms with this function.

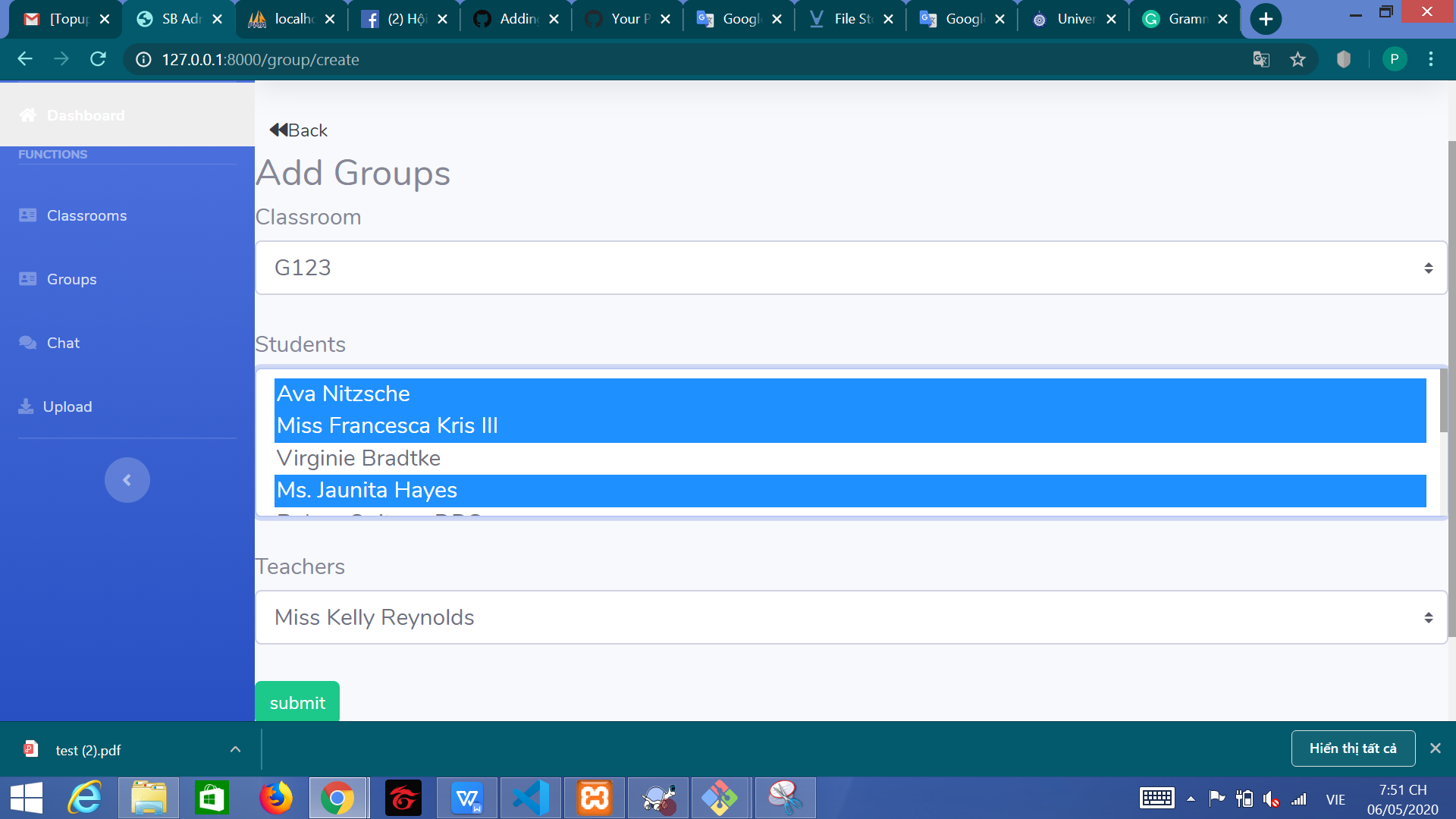
## 7.2. Groups



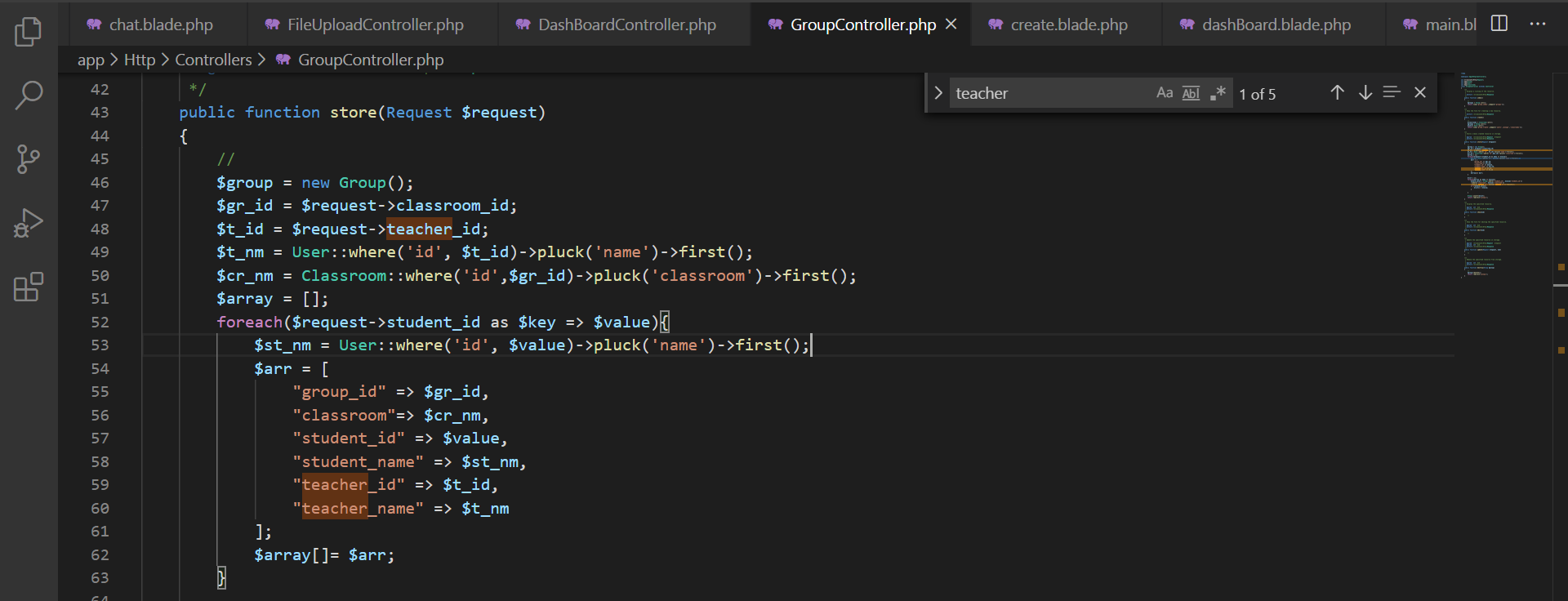


In this function, you can see the list of groups.





This function is Add Groups. The special thing in this function is that you can add multiple users. You can hold Ctrl and click on the users you want to add.

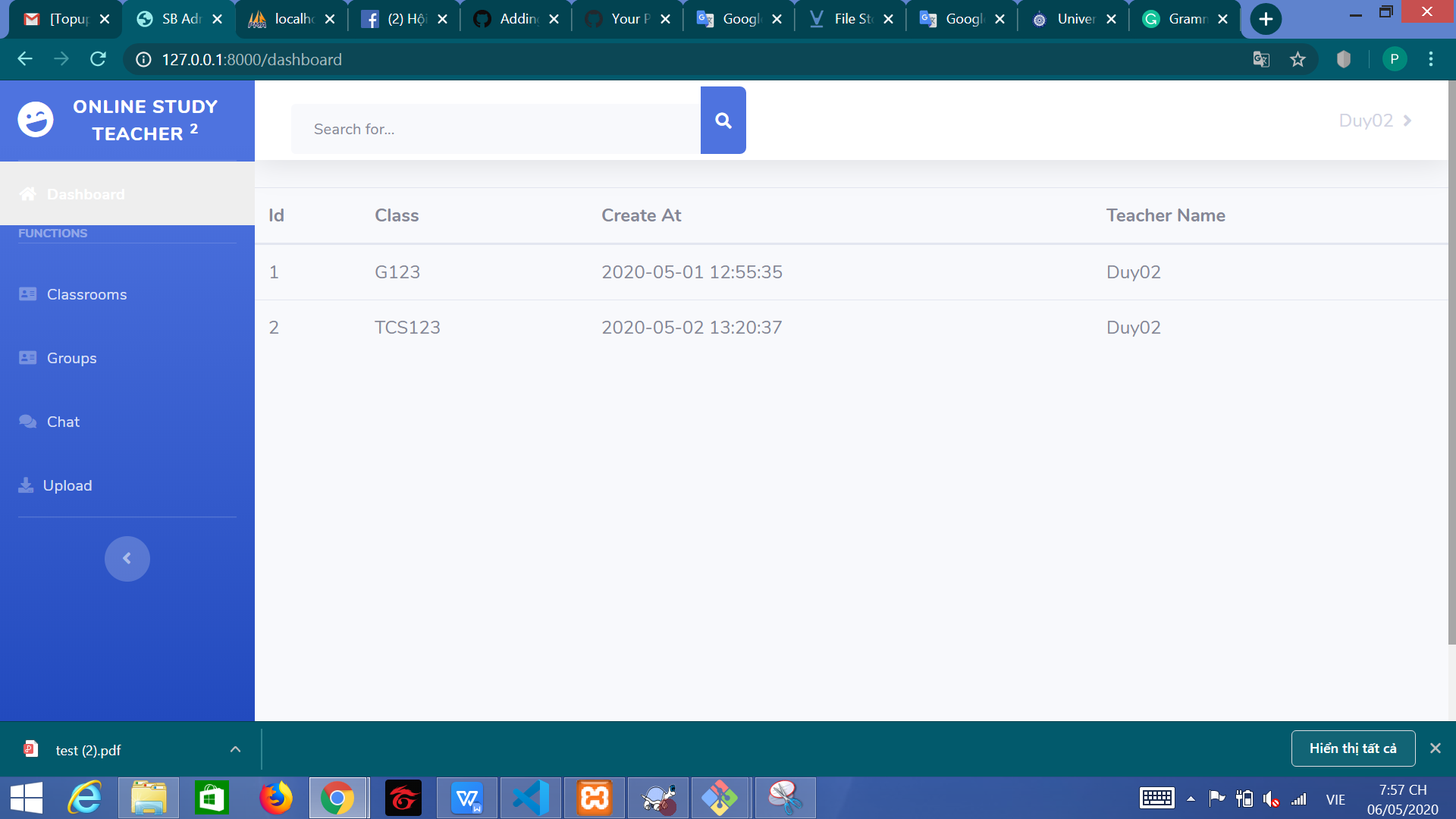


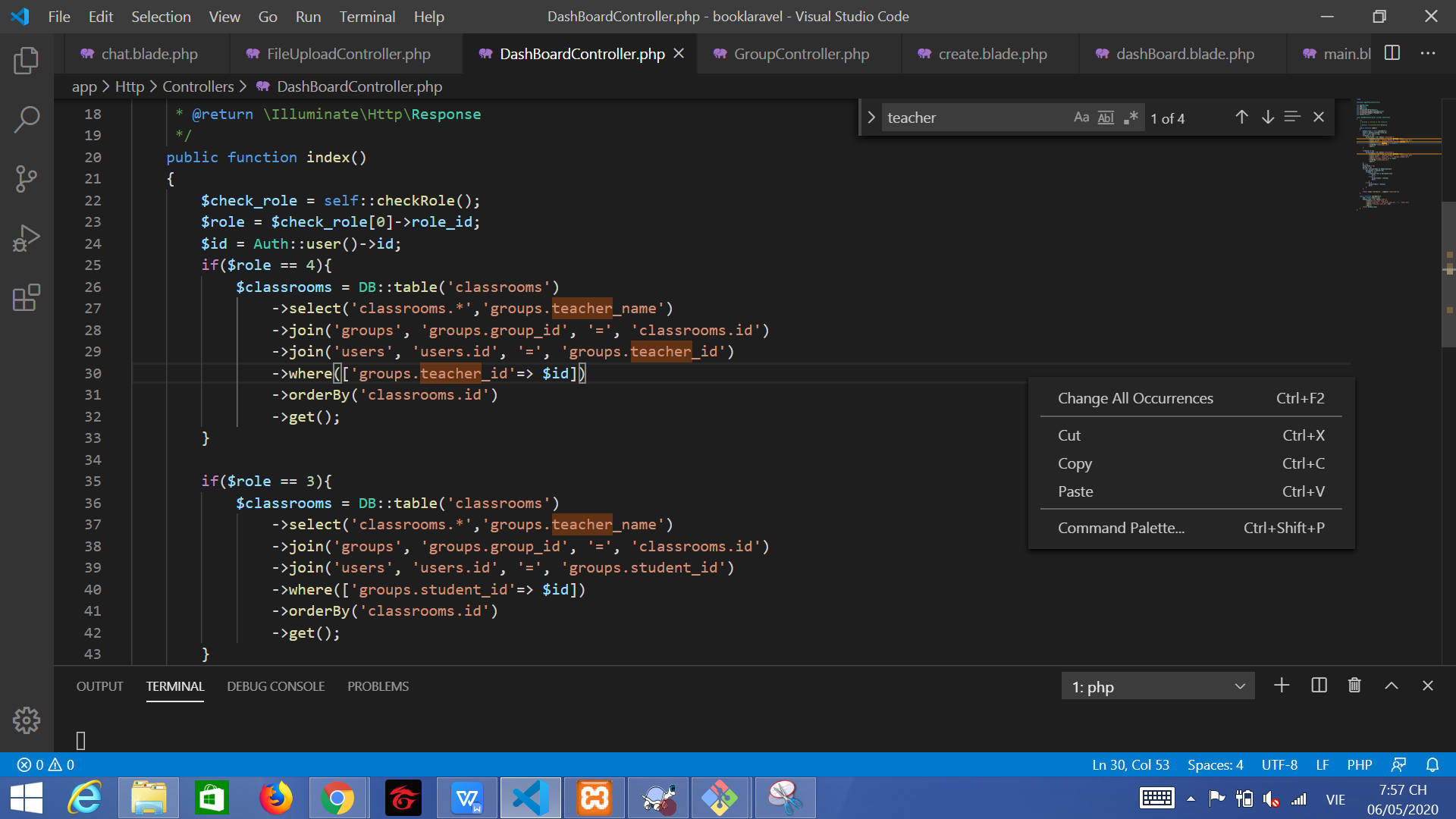
I have using foreach to insert multiple data. But If you choose user, you have add before. Don’t worry.



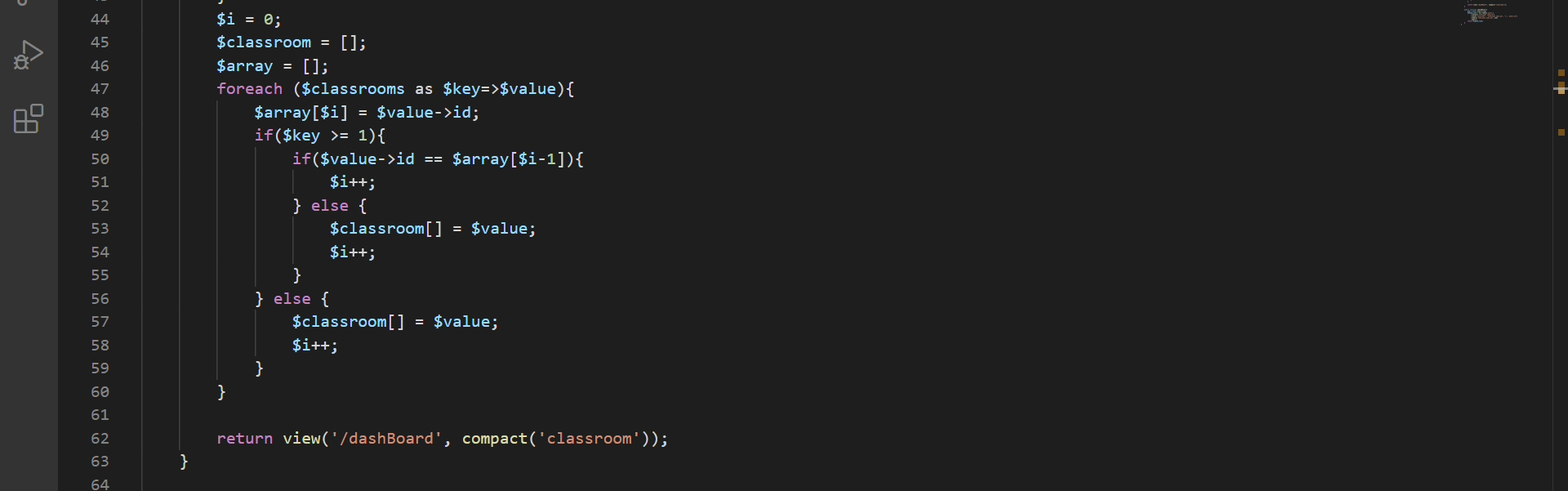
I have to check if user have add in database. This foreach have remove this users and not save in database.

## 7.3. Dashboard



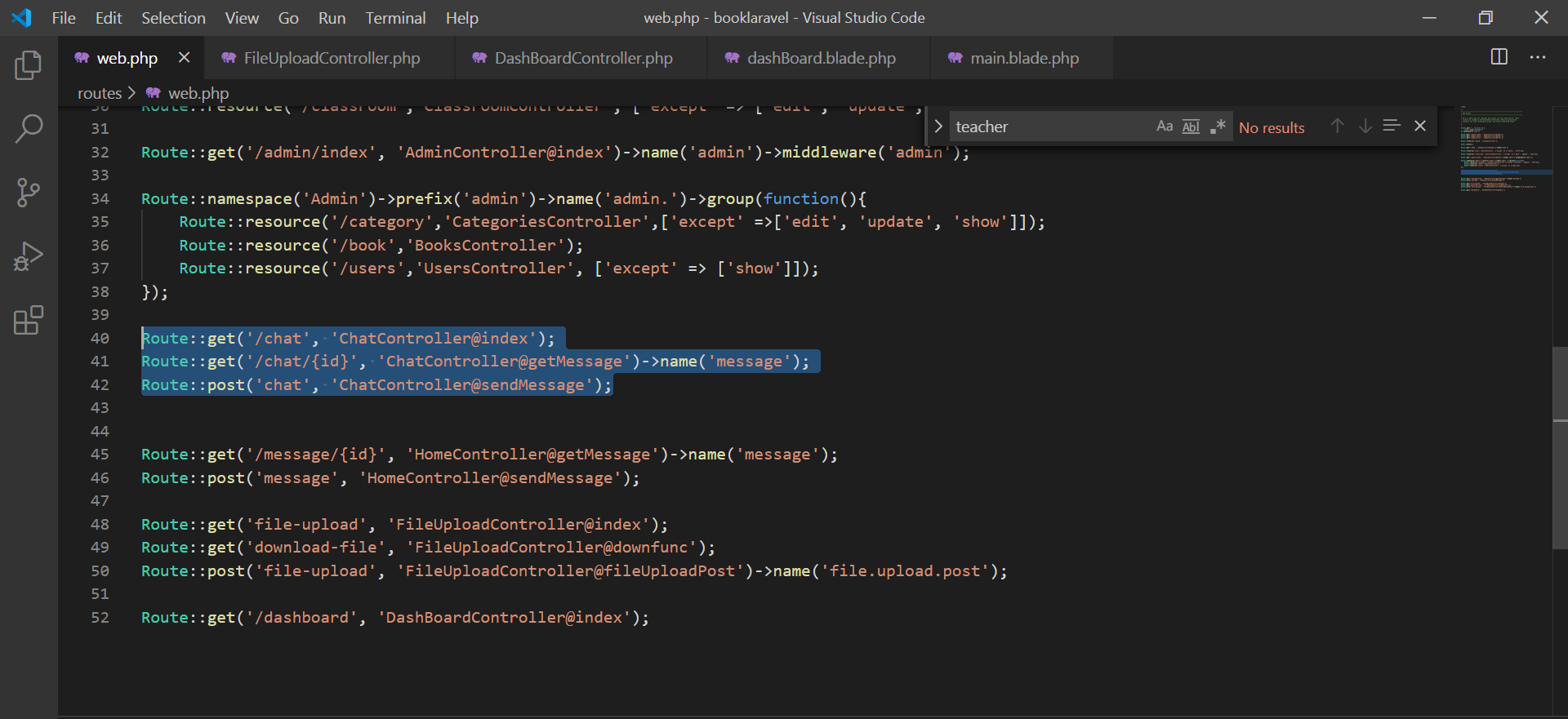


This function have get all class user have joins. But in teacher role, i have get same value a lot. I need to check the same data. This is the code check

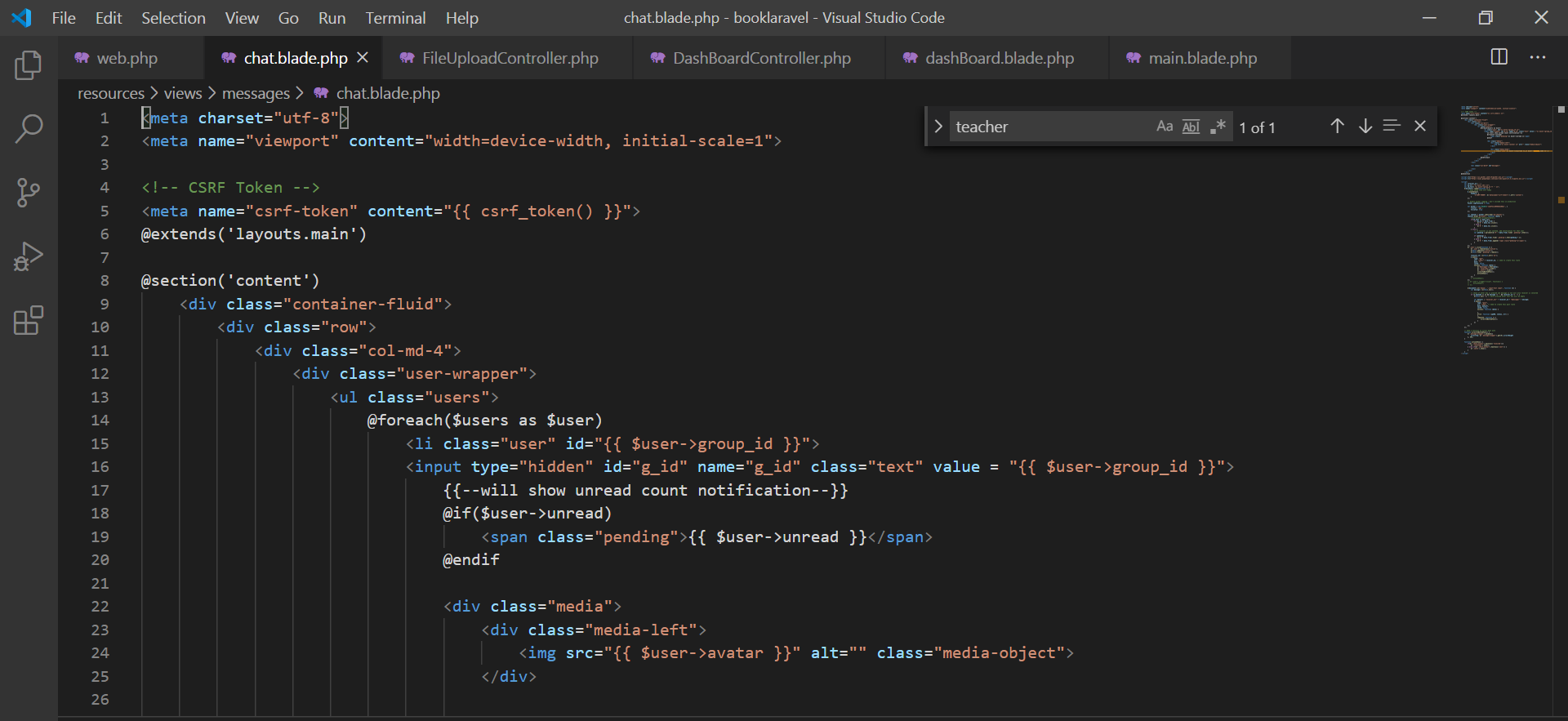


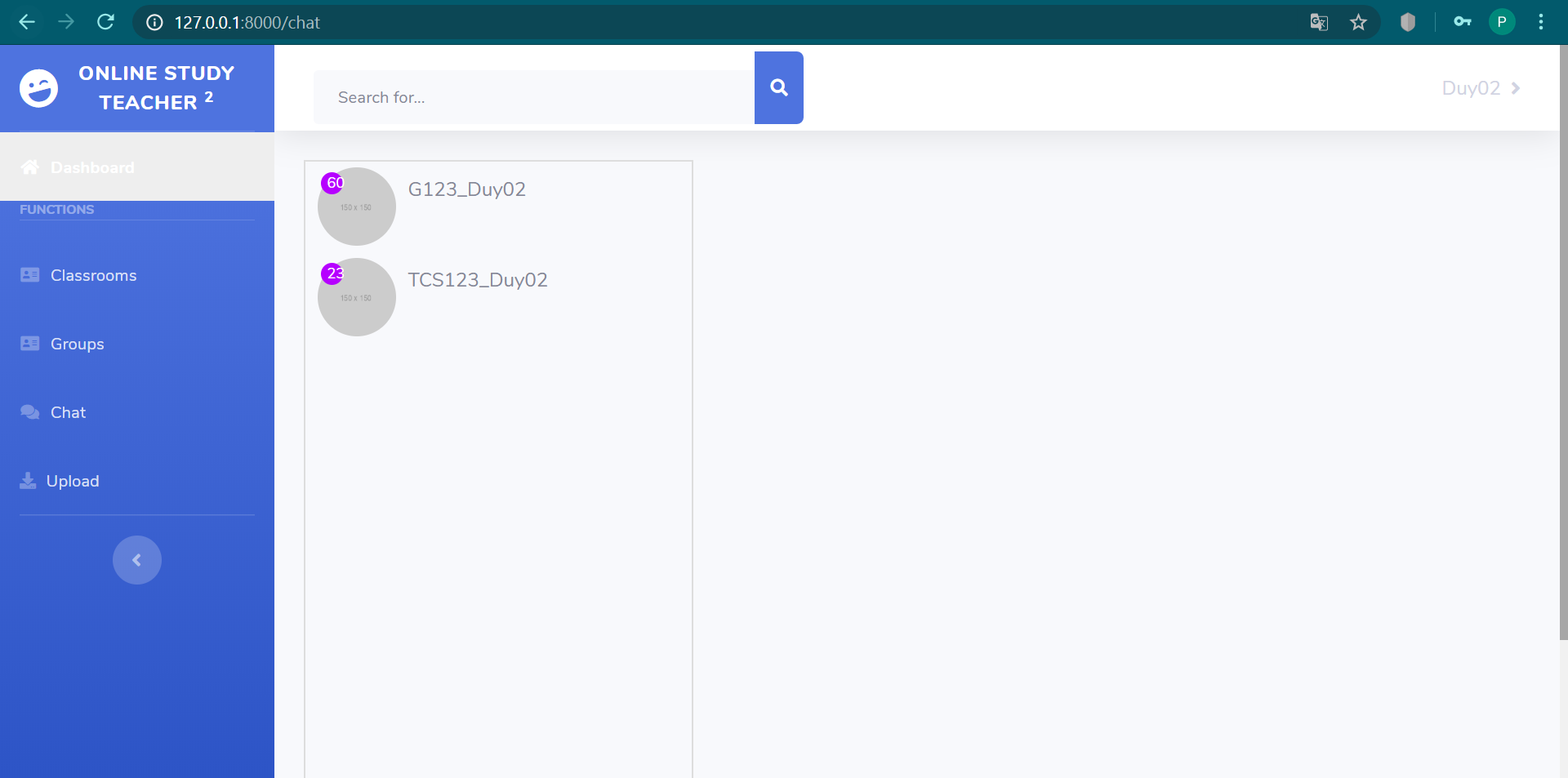
I have using the next variable in array and check before this variable.

## 7.4. Function Chat Groups

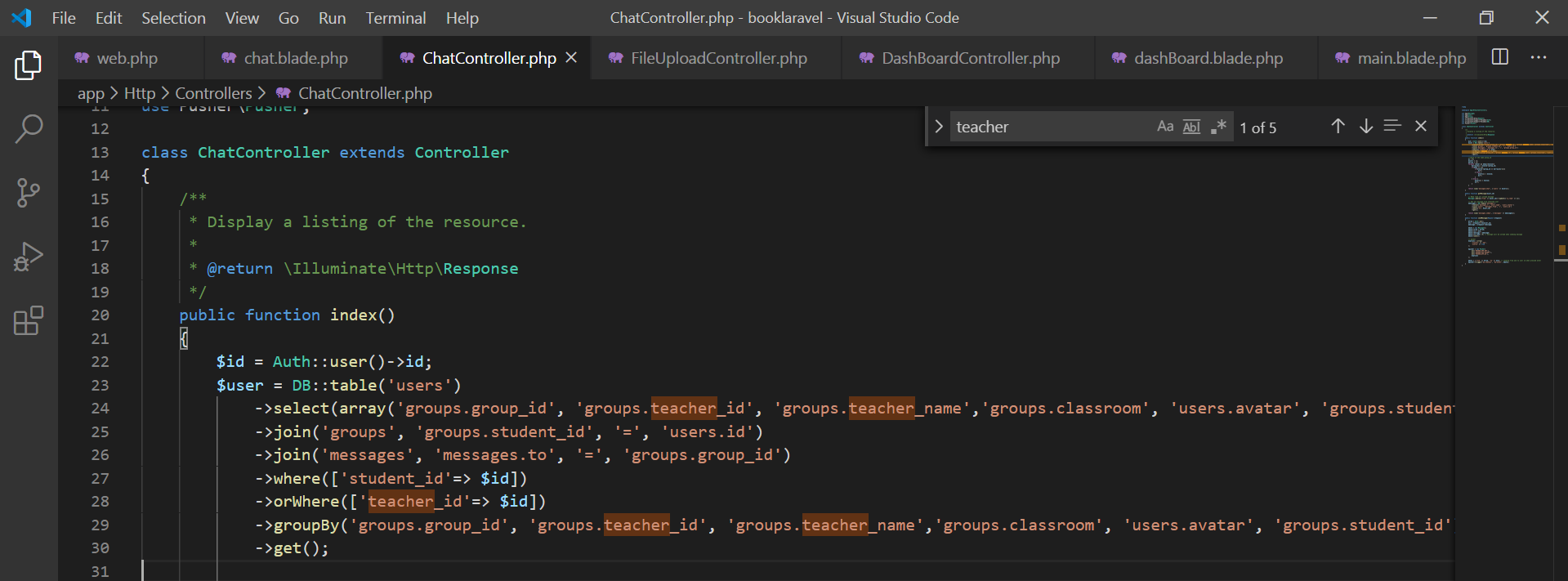


This is a route using in function Chat Groups

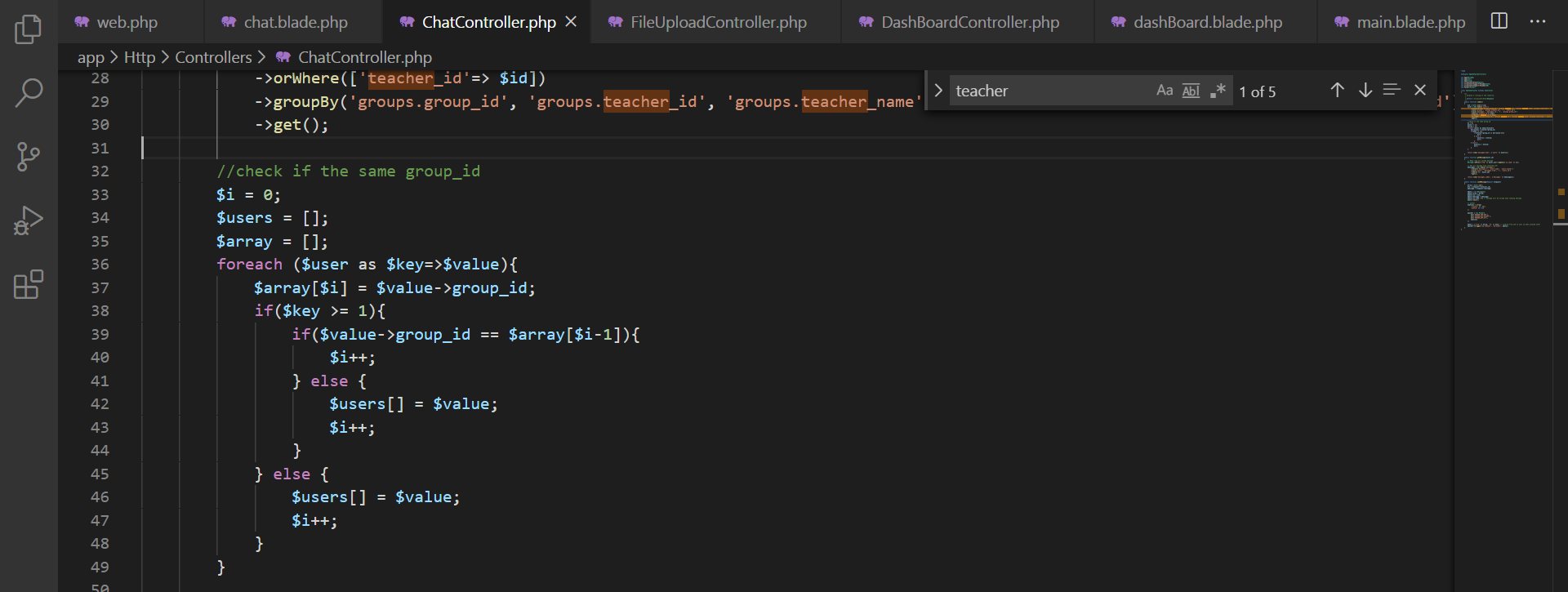




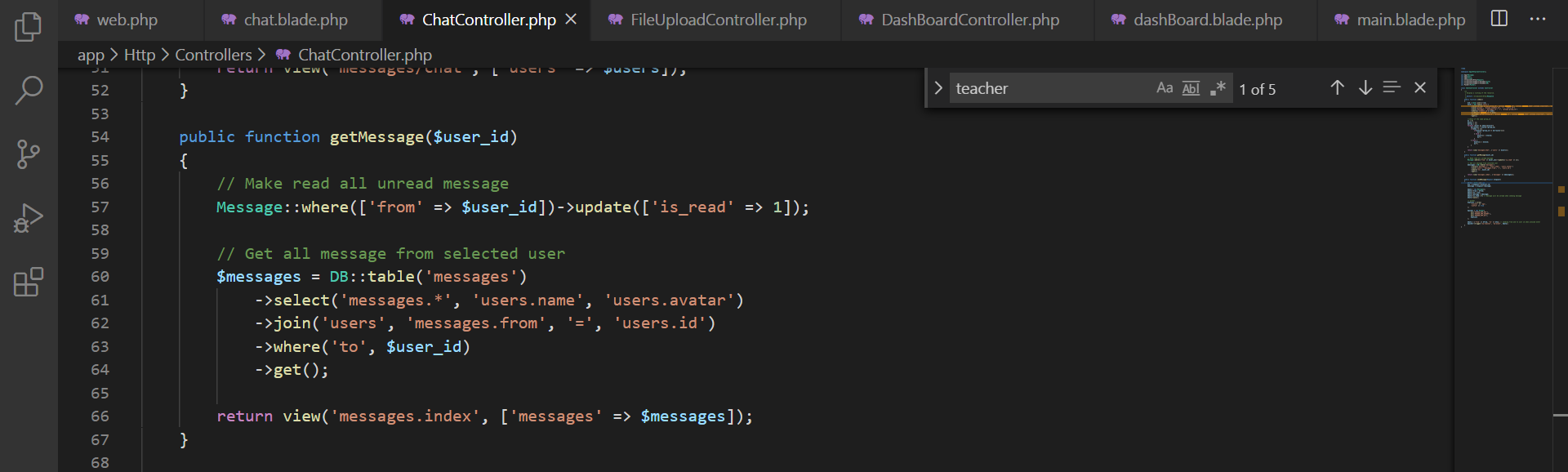
This is HTML show list groups you can have join in class

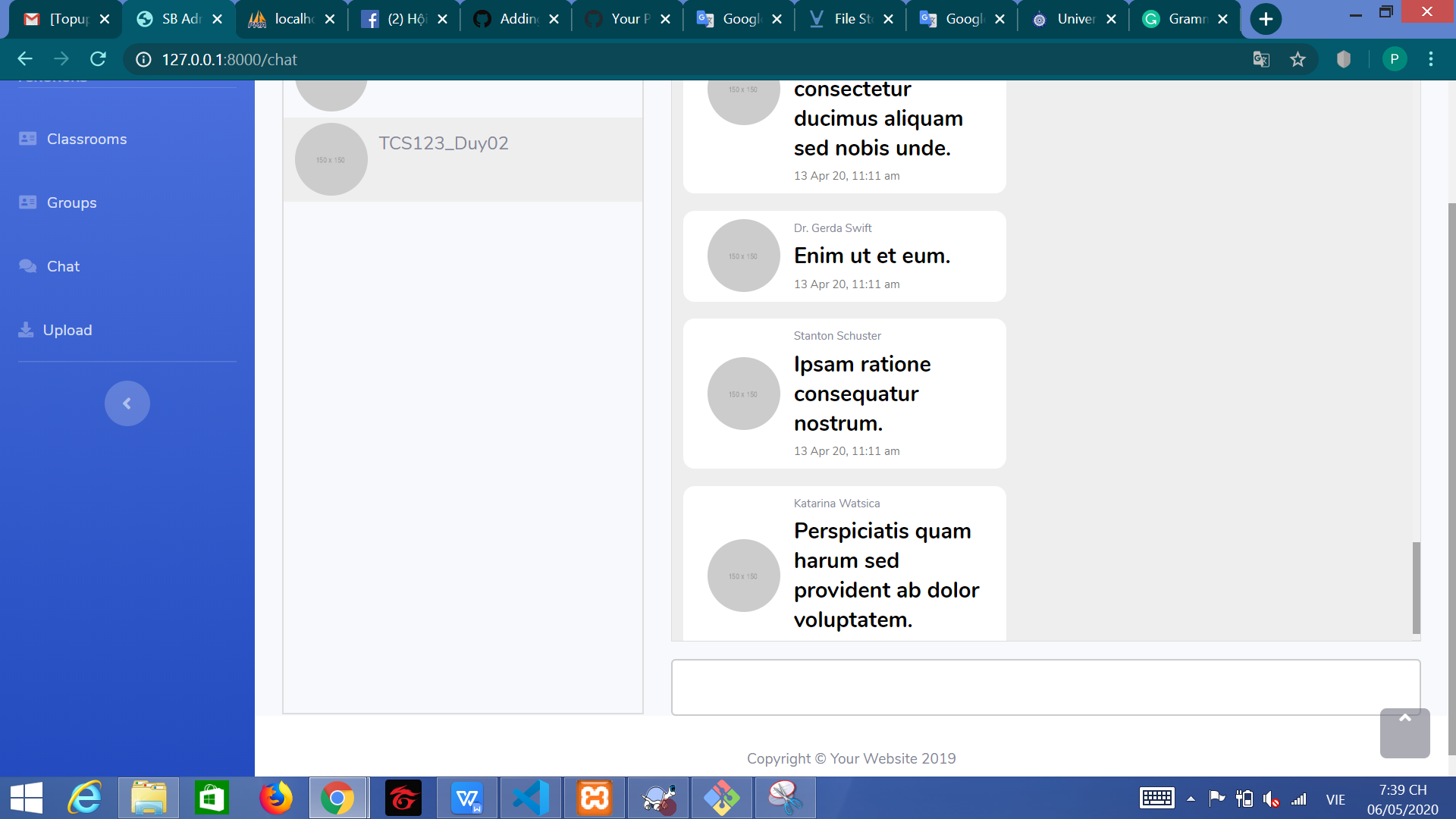


This is query select list Groups

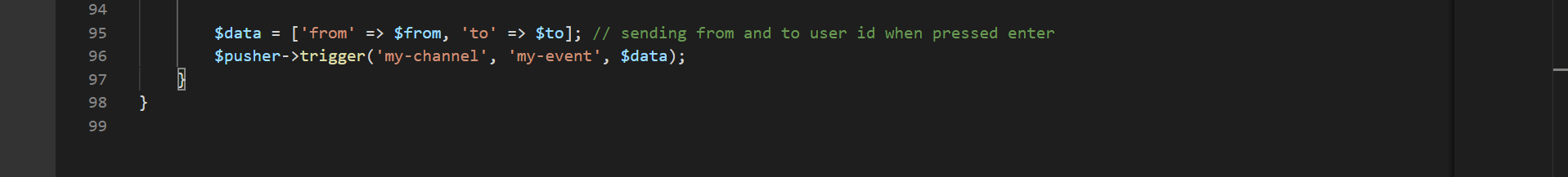
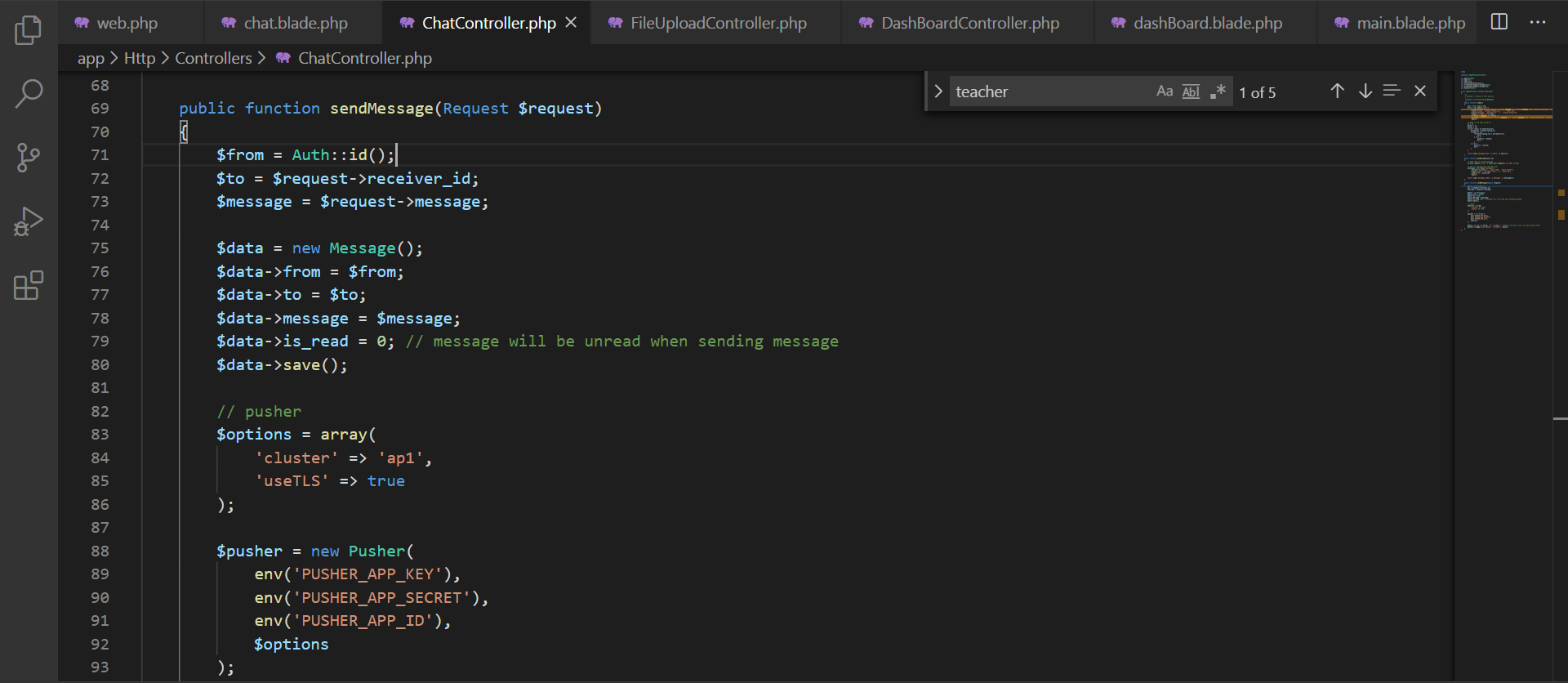


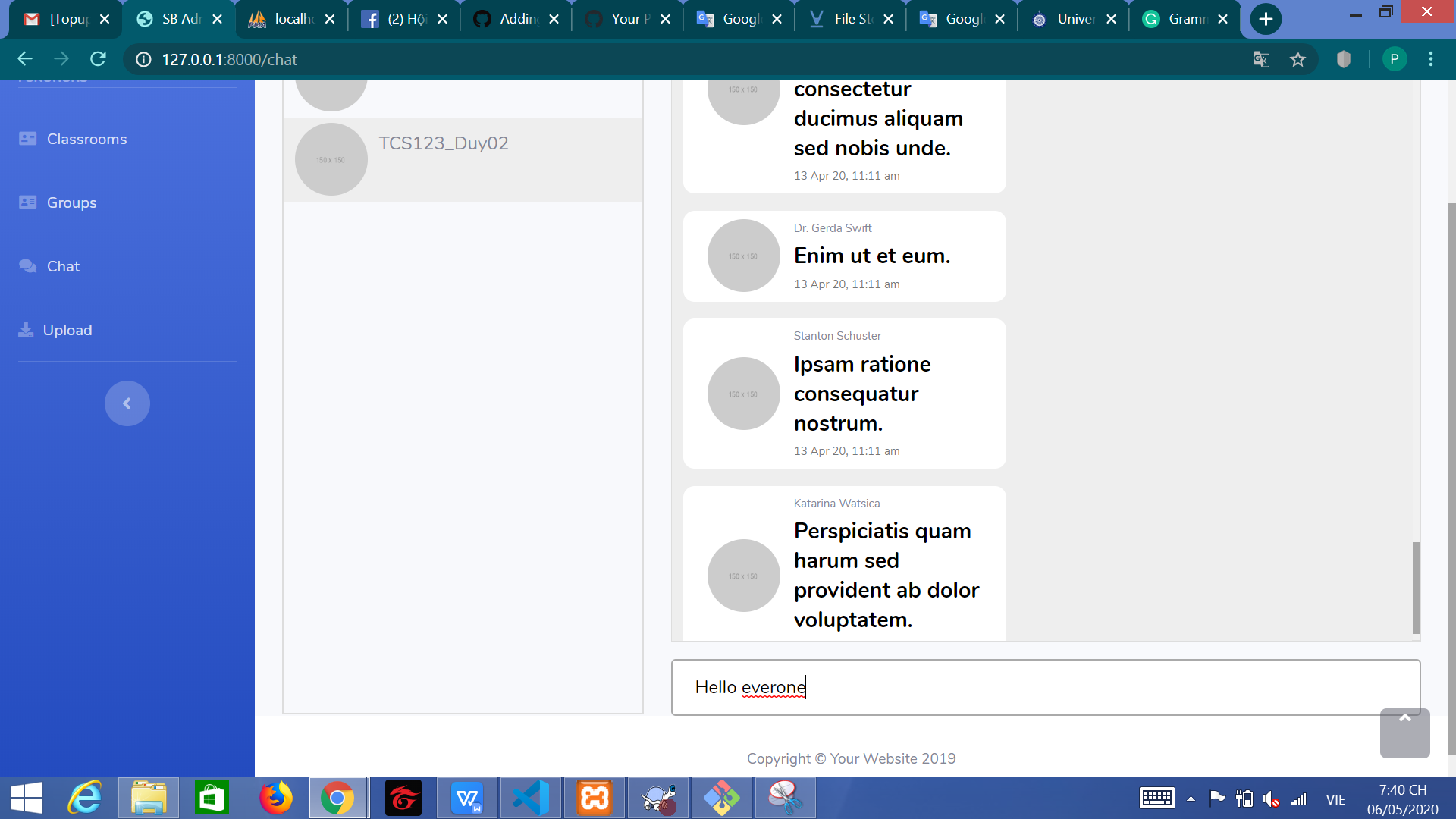
This is speacial code. This code know check the same value after they get value in query builder. This code i have check group\_id before group\_id in array

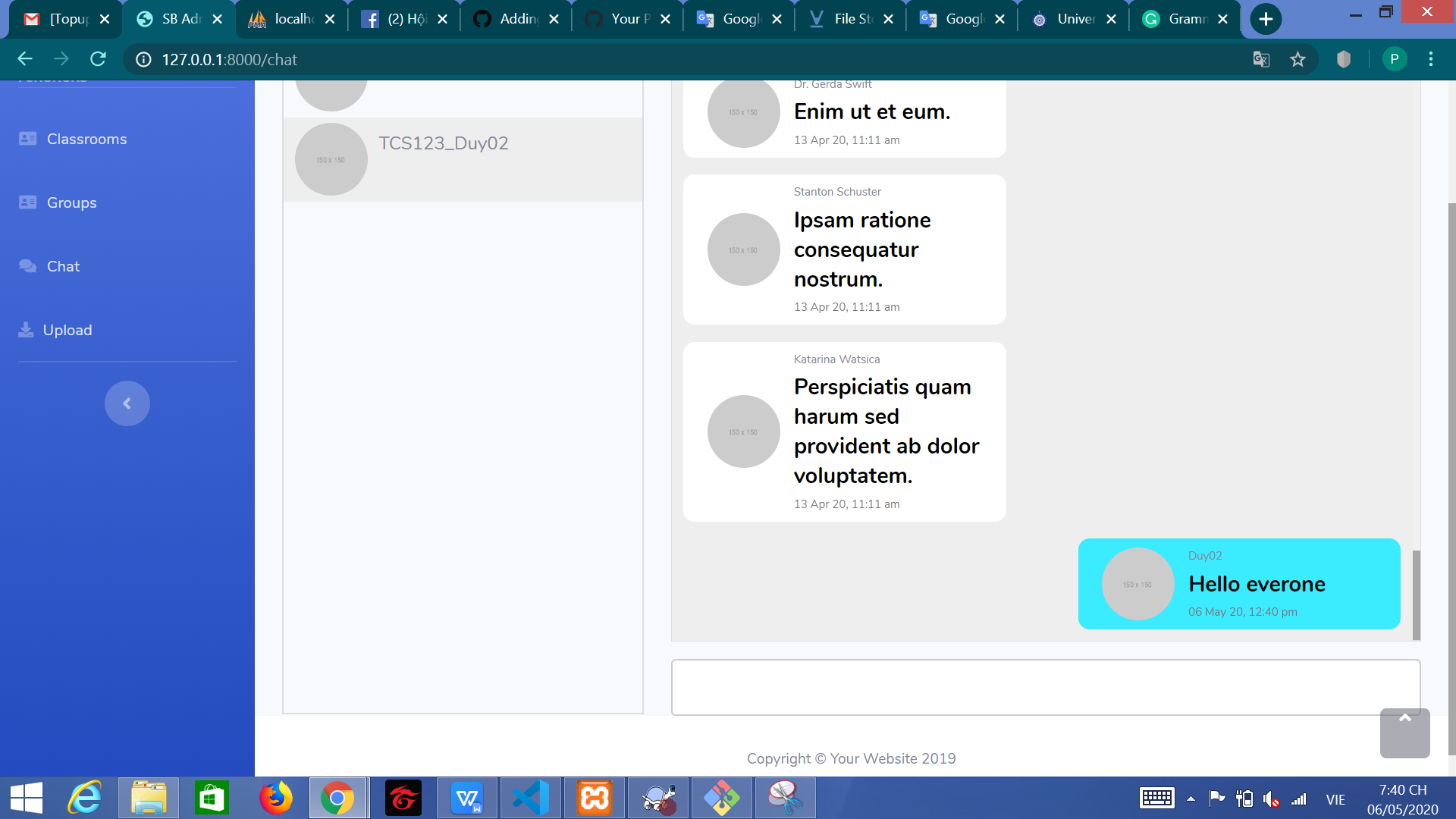




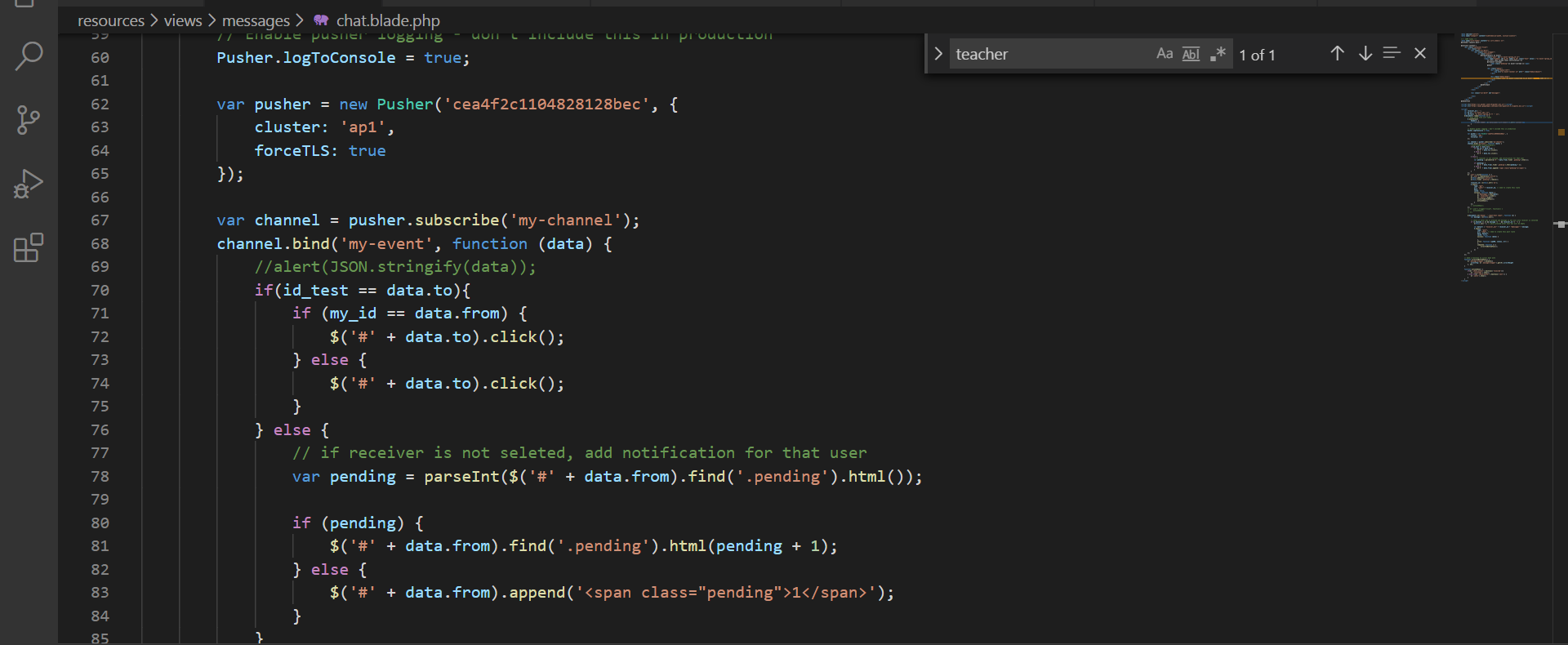
This code get message and show screen if you click chat box





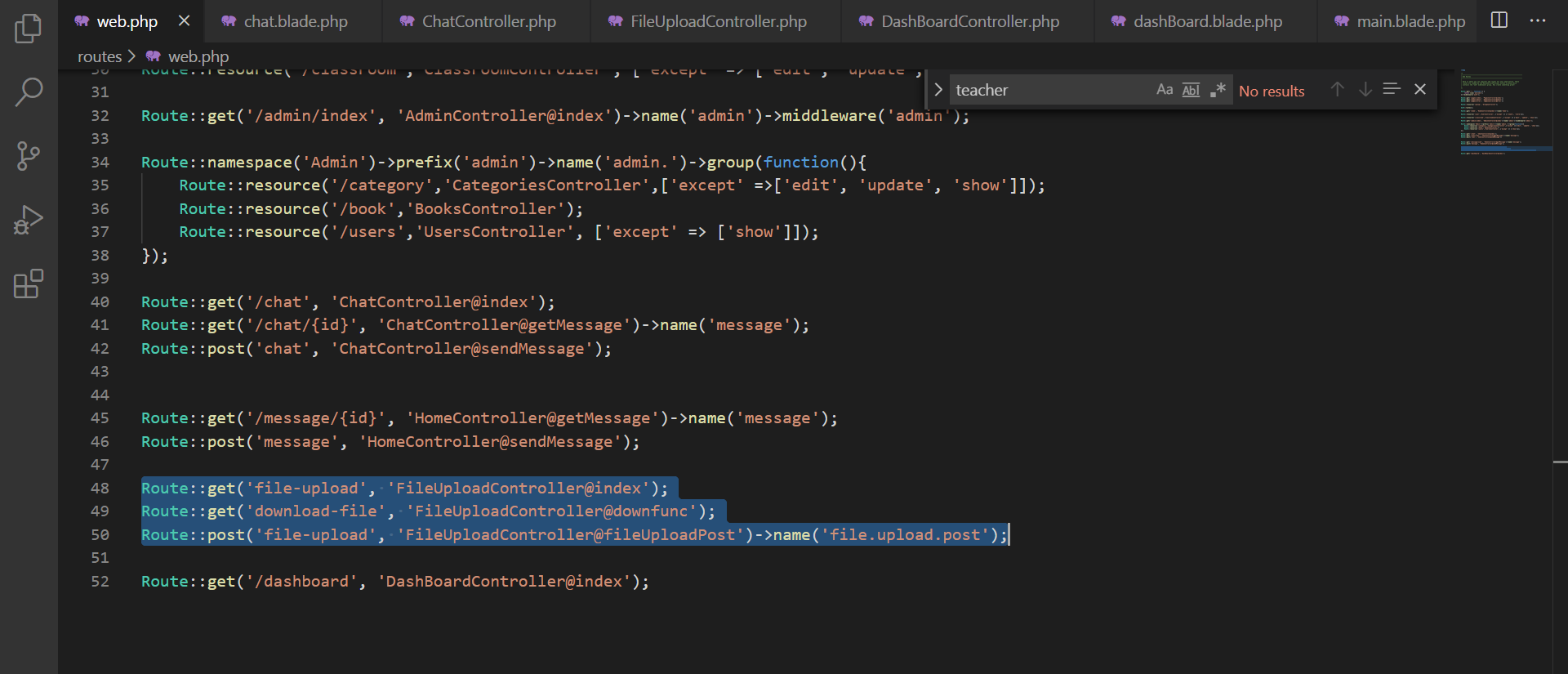


This code have send message. I using Pusher to call API and send message

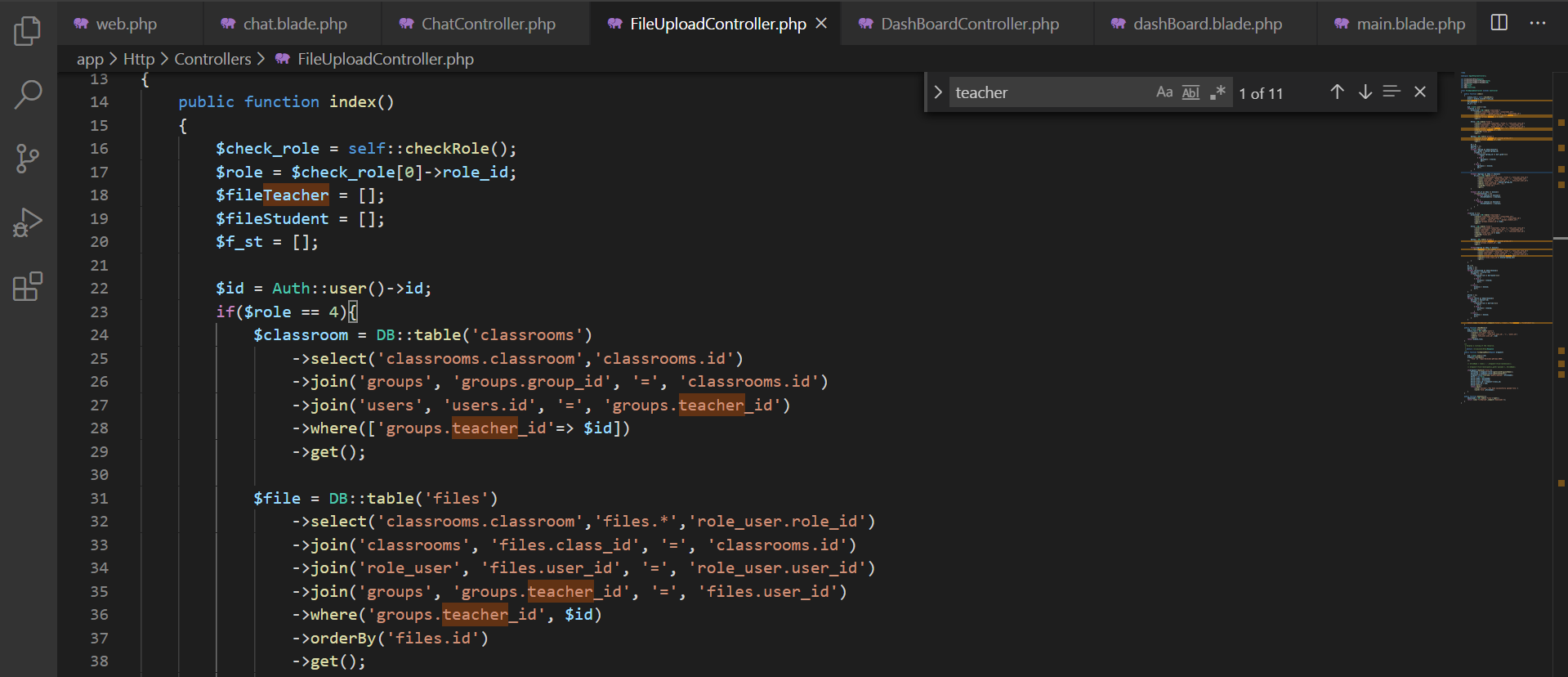


This code have show message when you sent message. I using Jquery in this stages. And this code will show a new messages

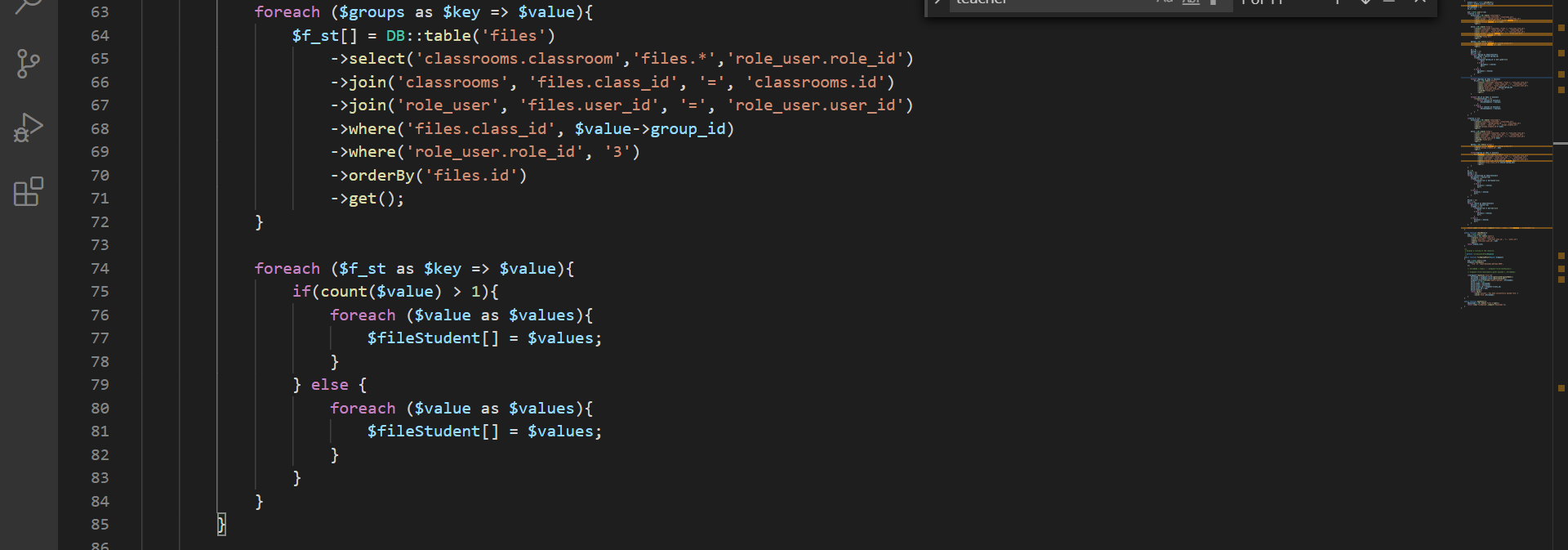
## 7.5. Upload Files

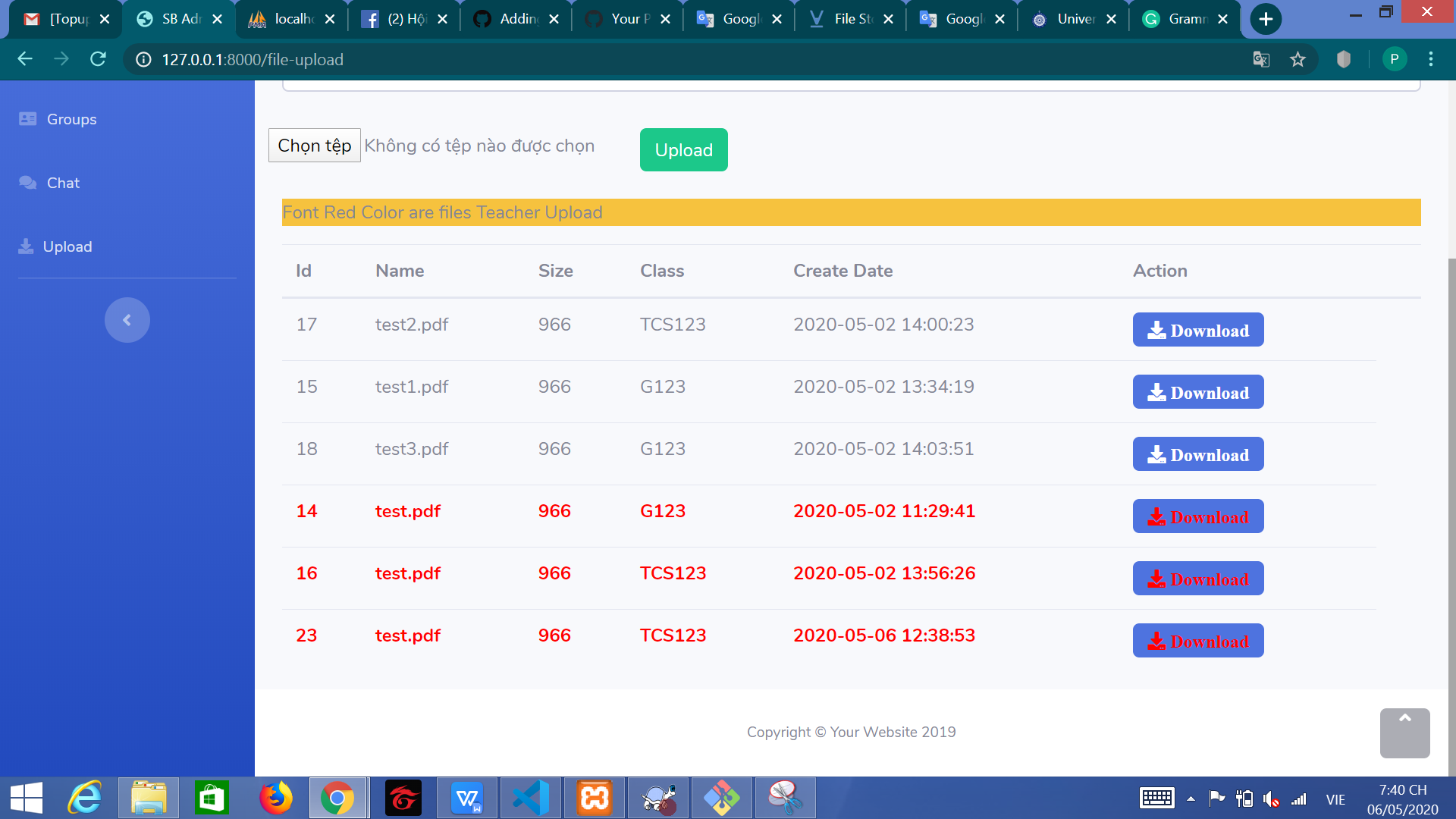


This is route of function Upload FIles

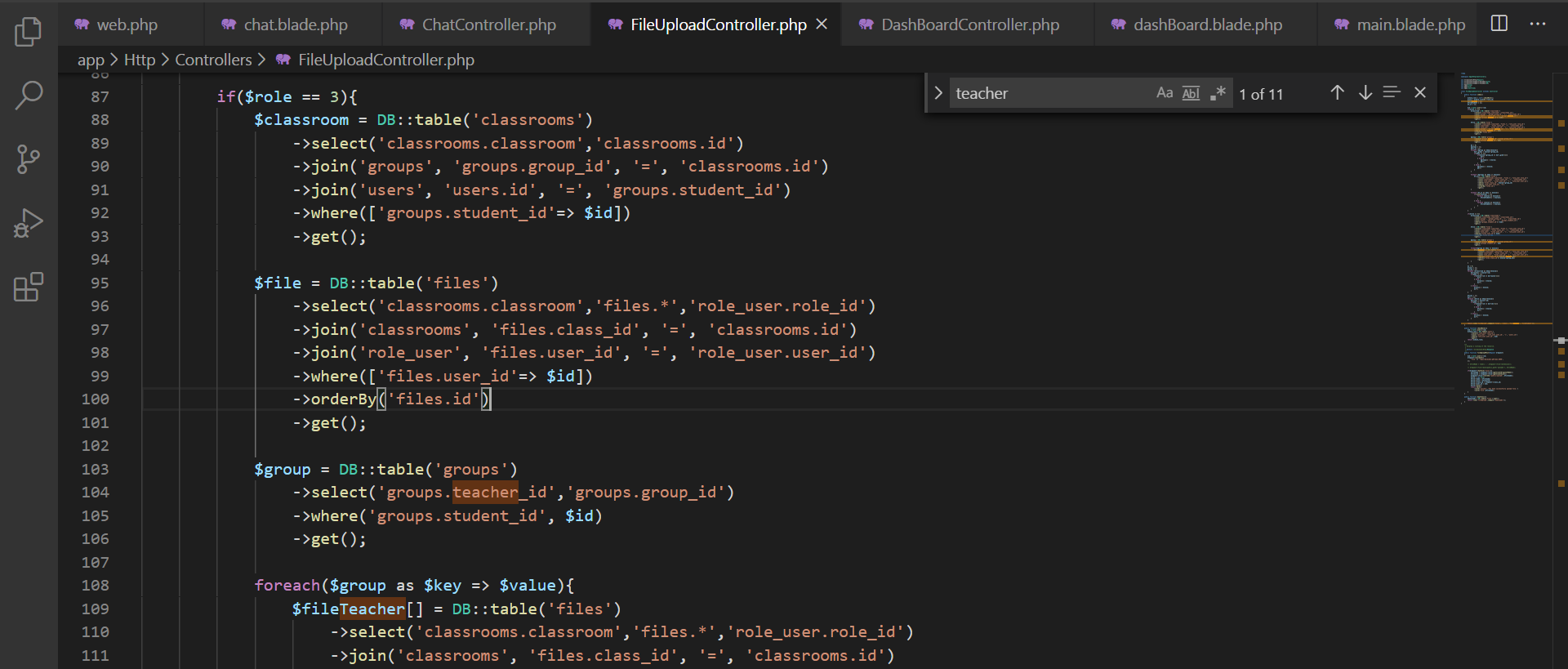


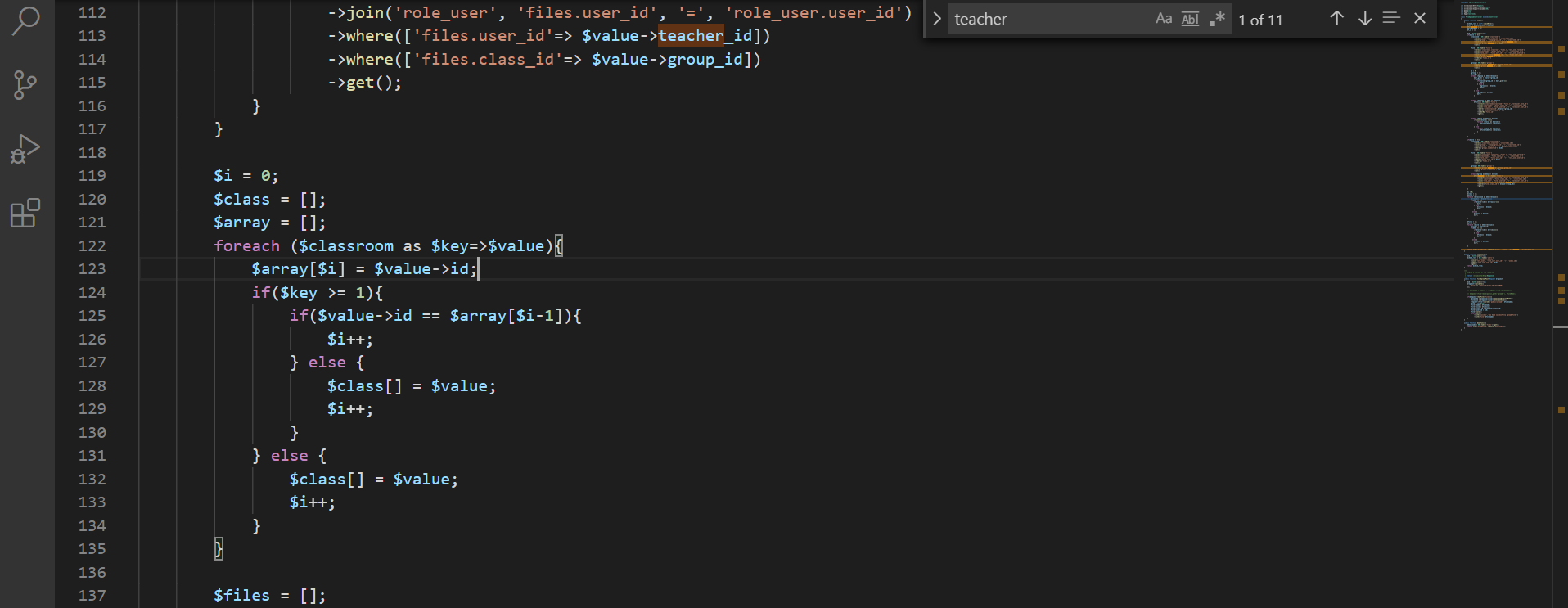


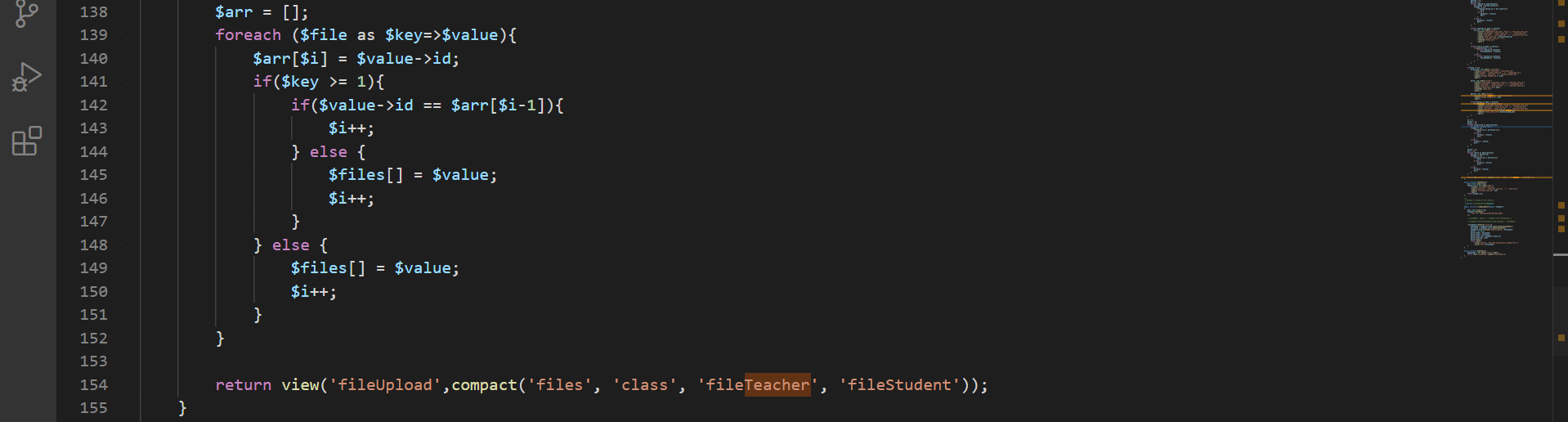




This is code i have show list download with role Teacher. This List Files have show of teacher files and all student files in teacher groups.



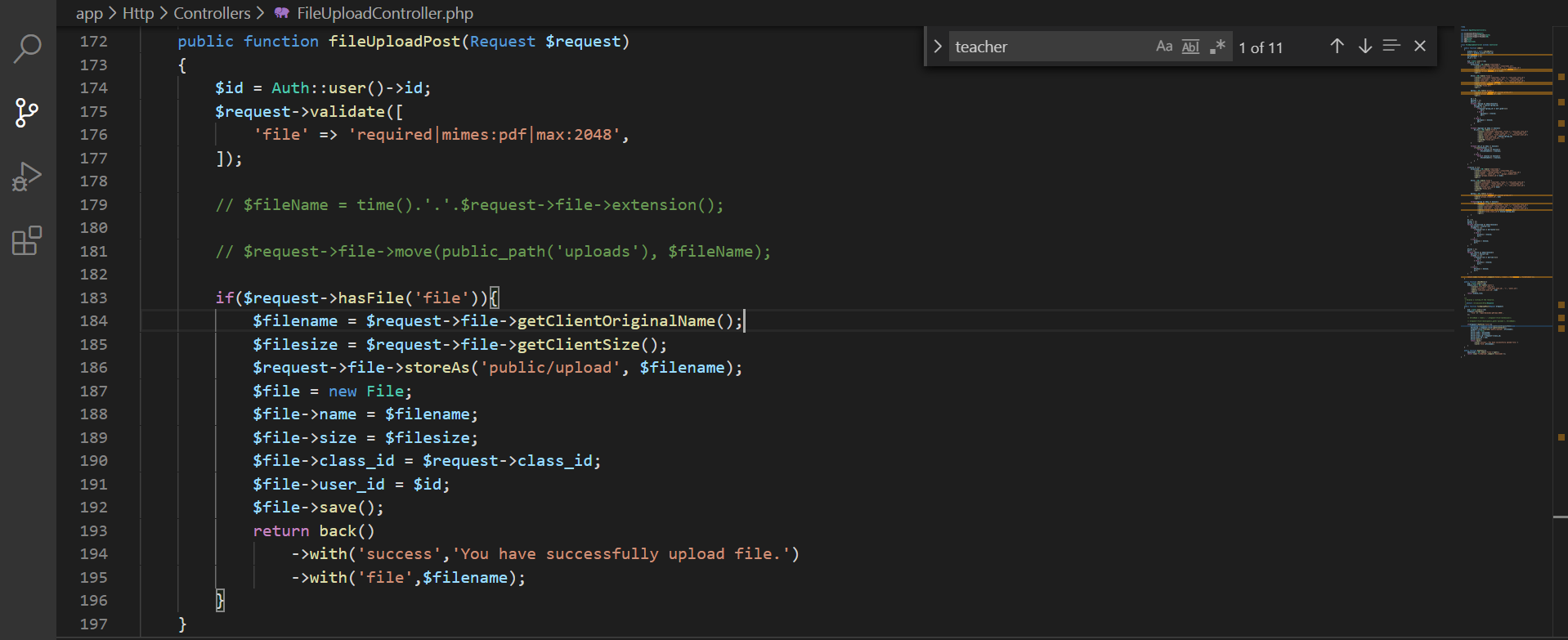


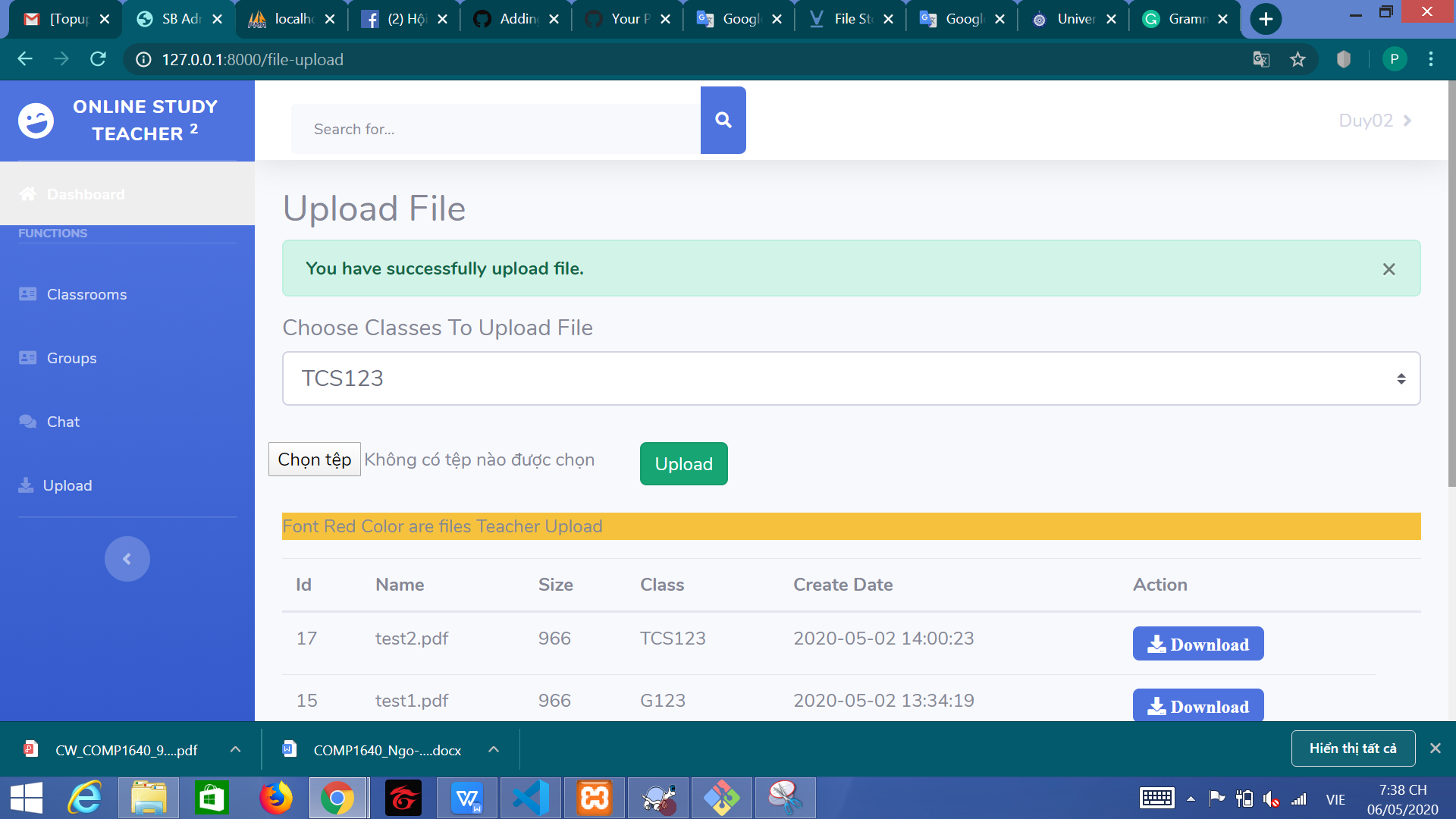


This is code i have show list with role student. In this code i have show files uploads of user when you login and files of teacher in your group you have join.



This is function check role user login.

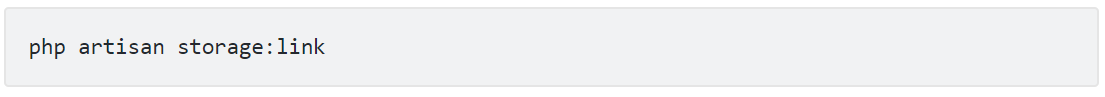




This function have upload files. This function i have check require file paf. I save the url and size of file

If upload file success, i will send message success

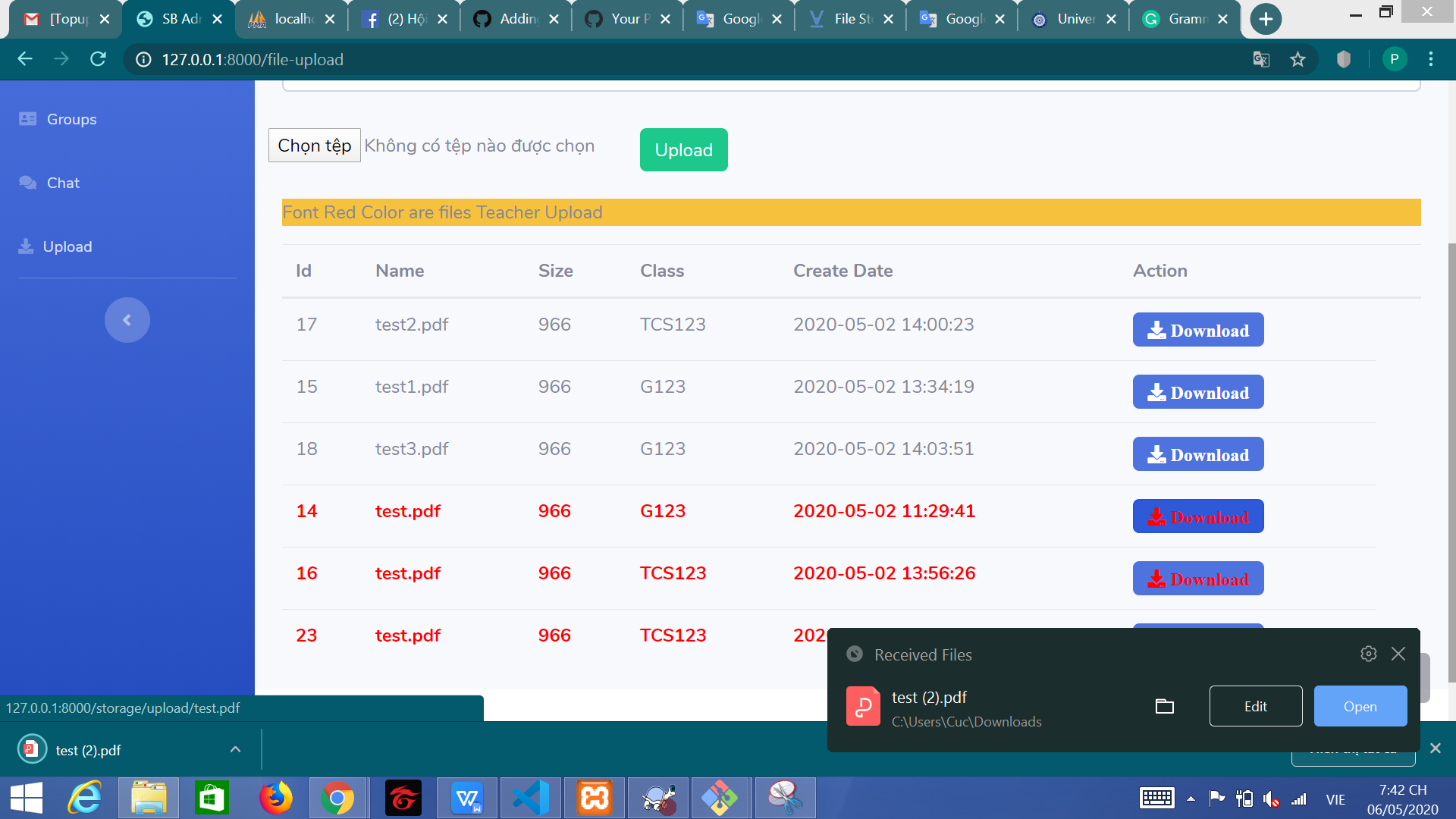
Ans special things in my source code. You need to run Artisan command when you want to download files:



This is the command that must be executed if you want to download files.







This is function have download file. I using Laravel Storage in this stages. If you want to download files, you need to run Artisan command:

# **Appendix**

*PHP* – short for Hypertext Preprocessor, is an open source scripting language. It is a widely-used language that is suitable for web development. It can be embedded into HTML. (Peter Cowburn, 2020)

*MySQL* – an open-source SQL database management system from Oracle Corporation. (MySQL, 2020)

*XAMPP* – a free Apache distribution containing MariaDB, PHP, and Perl. It is easy to install and use thanks to the setup made by Apache Friends. (Apache Friends, 2020)

*Lavarel* – a web application framework with elegant and expressive syntax. It aims to create an enjoyable and creative experience to development. (Lavarel, 2020)

*HTML* – short for Hyper Text Markup Language, is a computer language devised to allow website creation. It is powerful and easy to learn. (Ross Shannon, 2012)

*CSS* – short for Cascading Stylesheets, is a style sheet language. It is used for setting the look and feel of a website. (Robert Duncan, 2020)

*JavaScript* – a scripting or programming language that can be used for implementing complex features on web pages. When it comes to standard web technologies, it is the third layer of the ‘cake’, two of which are HTML and CSS. (MDN contributors, 2020)

# **TABLE OF FIGURES**

[Figure 1 ERD 7](#_Toc39740251)

# **TABLE OF REFERENCES**

Peter Cowburn (2020) *php.net* Available at: https://www.php.net/manual/en/intro-whatis.php (Accessed 7 May 2020)

MySQL (2020) *mysql.com* Available at: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html (Accessed 7 May 2020)

Apache Friends (2020) *apachefriends.org* Available at: <https://www.apachefriends.org/index.html> (Accessed 7 May 2020)

Lavarel (2020) *lavarel.com* Available at: <https://laravel.com/docs/4.2/introduction> (Accessed 7 May 2020)

Ross Shannon (2012) *yourhtmlsource.com* Available at: <https://www.yourhtmlsource.com/starthere/whatishtml.html> (Accessed 7 May 2020)

Robert Duncan (2020) *simplehtmlguide.com* Available at: <http://www.simplehtmlguide.com/whatiscss.php> (Accessed 7 May 2020)

MDN contributors (2020) *mozilla.org* Available at: <https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript> (Accessed 7 May 2020)