

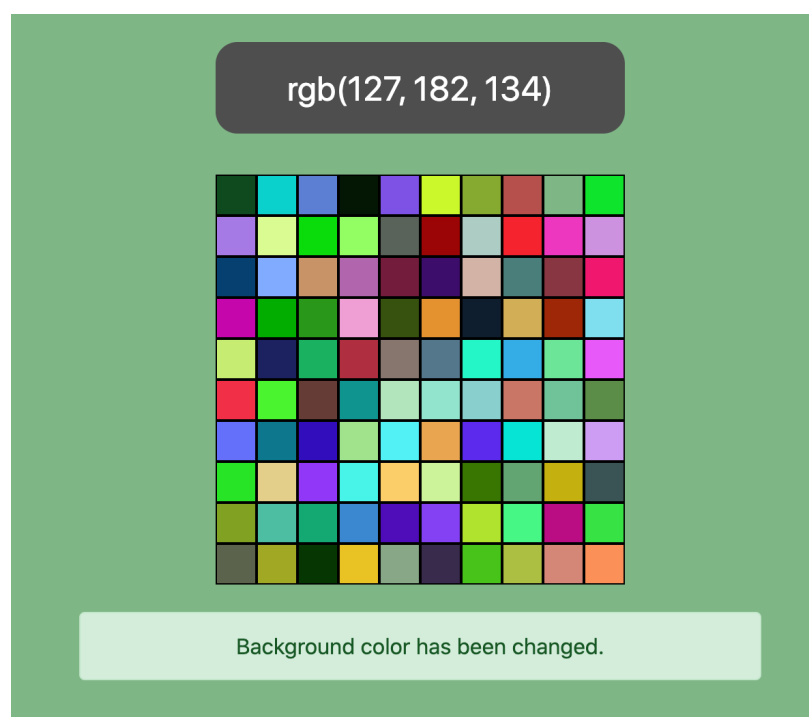
WEB PROGRAMMING AND APPLICATIONS

(503073)

WEEK 5

Prepared by Mai Van Manh

Exercise 1: Use **jQuery** to design a color picker web page like the one below.



See the video demo (*Ex1 - Color picker.mp4*) to better understand the requirements of this exercise.

Requirements:

- The cells must be generated by **Javascript/jQuery** (not hardcoding by HTML).
- The cell colors must be generated **randomly** and will be changed every time the page is reloaded.
- You are free to choose which colors to display.
- When mouse overs the cells, the background color changes temporarily according to the value of the cell and will return to the original color when the mouse is moved out of the color panel.

When you click on a cell, the background color changes to the color of the cell. Messages will show and hide in 3 seconds with fade in and fade out effects.

Exercise 2: Solve this exercise by using **jQuery** instead of regular Javascript:

User Login

Email:

Password:

You need to write javascript code to validate the form's information when user click the submit button. When the **submit** button is clicked, the following conditions must be checked:

- All fields should be fulfilled.
- A valid email should be entered.
- Password must contain at least 6 characters.

The conditions must be validated in the order listed above. If one of the conditions is failed, you should stop the validation and display the corresponding error message immediately. Here are some examples of the error messages:

- *"Please enter your email"*.
- *"Your email is not correct"*.
- *"Please enter your password"*.
- *"Your password must contain at least 6 characters"*.

When showing the error message, the corresponding text field should be focused and then if the user click on text field, the error message should be disappeared.

User Login

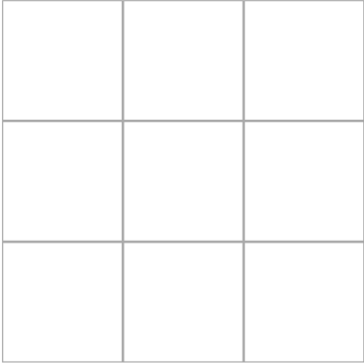
Email:

Password:


Please enter your email address.


Exercise 3: Use **jQuery** to create puzzle games like the following:


Xếp hình



Xếp hình







Congratulations! You have completed the puzzle game.

See the video demo (*Ex3 - Puzze-example.mp4*) to better understand the requirements of this exercise.

Requirements:

- The position of the puzzle pieces is chosen randomly when the page is loaded. Rotation angles (0, 90, 180, 270) of the puzzle pieces are also randomly generated.
- Users can click on the pieces to rotate them (0 → 90, 90 → 180,...).
- You can only drag pieces into empty box, you can not drop pieces in the none empty box or outside.
- After the user drags the last piece of the puzzle, the web page will display a greeting message if the user has correctly cropped the original photo.

Exercise 4: There is a student list stored in a MySQL database. A student's information includes: code, name, email, phone number. You are provided APIs to manipulate on this student list. The API list includes actions:

- Get student list
- Add a new student
- Delete a student
- Update a student's information

You don't need to care about how the data is stored in the database. You just need to build the website, call the API to send the data to the server for the server to process and receive the results. The data returned from the server comes in two forms: **JSON Array** if it is a student list or **boolean** if it is the result of add / delete / edit action.

Your task is to build a website as shown below.

Student Management using Ajax

Name:

Enter name

Email:

Enter email

Phone:

Enter phone

Add

Update

ID	Name	Email	Phone	Action
1	Lam Truong	john@example.com	01234567789	<div>Edit</div> <div>Delete</div>
2	Cam Ly	john@example.com	01234567789	<div>Edit</div> <div>Delete</div>
3	My Tam	john@example.com	01234567789	<div>Edit</div> <div>Delete</div>

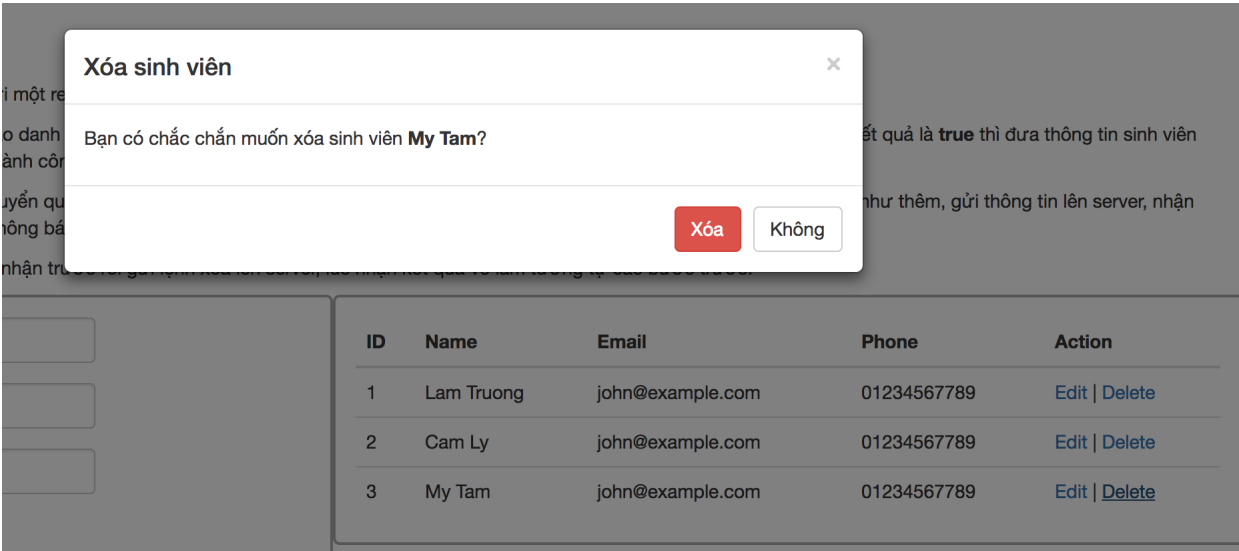
Success! Delete student success.

Failed! An unknown error occured. Please try again later.

Requirements:

- Use the combination bootstrap and jquery to complete this exercise.
- Use jQuery post request to call APIs.
- First, you need to send a request to the server to receive a list of students and list the students in the table.
- To add a new student, the user fills the form and clicks the **Add** button. At this time, student's information will be posted to the server through the API, the server will save the data to the database and return the result to you. If the result is successful, add a new student to the end of the display table and show the success message. If the result is unsuccessful then an error message will be displayed to the user.

- Messages (success or failure) are displayed slowly (fade In) and will be hidden slowly (fade out) after **3 seconds**.
- When the user presses the **Edit** button, the corresponding student information is filled in the form for editing. **Now the Update button will be enabled and the Add button will be disabled**. If the user clicks the **Update** button to save, the process of sending the data to the server and displaying the results will work similarly. **Finally, restore the initial state of the Add button and Update button.**
 - Removing a student also works the same way as adding, and editing, students. However, before deleting, you need to display a confirmation dialog box (**use jquery modal dialog**).



API Description:

1. Get all student
 - a. File name: get-students.php
 - b. Method: GET
 - c. Parameters: None
 - d. Result:

Success	Failed
<pre>{ "status": true, "data": [{ "id": "1", "name": "Nguyen Minh Tien", "email": "minhtien@gmail.com", "phone": "01292101010" }, { "id": "2", "name": "Cao Minh Toan", "email": "minhtan@gmail.com", "phone": "0987223221" }] }</pre>	<pre>{ "status": false, "data": "An error occurred." }</pre>

2. Add new student:

- a. File name: add-student.php
- b. Method: POST
- c. Parameters: name (string), email (string), phone (string).
- d. Result:

Success	Failed
{"status":true,"data":"Add students success"}	{"status":false,"data":"An error occurred."}

3. Delete a student:

- a. File name: delete-student.php
- b. Method: POST
- c. Parameters: id (int)
- d. Result:

Success	Failed
{"status":true,"data":"Delete students success"}	{"status":false,"data":"Student not found."}

4. Update a student

- a. File name: update-student.php
- b. Method: POST
- c. Parameters: id (int), name (string), email (string), phone (string).
- d. Result:

Success	Failed
{"status":true,"data":"Update students success"}	{"status":false,"data":"Student not found."}